Київський національний університет
імені Тараса Шевченка
Факультет комп'ютерних наук та кібернетики

Звіт до лабораторної №2
на тему: "NoSQL"
з дисципліни "Сучасні технології баз даних"

Виконала студентка 3-го курсу
Групи МІ-31
Спеціальності "Комп'ютерні науки"
Лагода Катерина Анатоліївна

Київ-2024

Види NoSQL баз

1) **Документоорієнтовані**:

- Дані зберігаються у вигляді документів (JSON, BSON, XML);
- Ідеально підходять для роботи з напівструктурованими даними.
- **Приклади:** MongoDB, Couchbase, RavenDB.

2) Ключ-значення:

- Дані зберігаються як пари "ключ-значення".
- Підходять для кешування та зберігання простих даних.
- **Приклади:** Redis, DynamoDB, Riak.

3) Графові бази даних:

- Дані зберігаються у вигляді вузлів (nodes) і зв'язків (edges), що утворюють граф.
- Застосовуються для аналізу зв'язків і взаємодій (наприклад, соціальні мережі).
- **Приклади:** Neo4j, ArangoDB, JanusGraph.

4) Стовпцеві бази даних:

- Дані зберігаються у вигляді стовпців, організованих у сімейства стовпців.
- Підходять для аналітичних задач та роботи з великими масивами даних.
- **Приклади:** Apache Cassandra, HBase, ScyllaDB.

5) Мульти-модельні бази даних:

- Поєднують кілька моделей (наприклад, документи, графи, ключ-значення) в одній системі.
- **Приклади:** ArangoDB, OrientBD

Винесення частини даних в NoSQL базу

Враховуючи мою предметну область – видавництво книжок, в NoSQL базу MongoDB буде доцільним винести, наприклад, наступні таблиці:

1) **reviews (Огляди книг)**

   Чому доцільно:

- Структура оглядів може бути різною (наприклад, додавання зображень, емоцій, тегів, або оцінок за різними критеріями).
- Дані часто читаються разом із книгами, тому зберігання оглядів як вкладених документів у колекції books MongoDB може покращити швидкість доступу.

2) **customers та orders (Замовлення клієнтів)**

   Чому доцільно:

- Дані про клієнтів та їх замовлення можуть бути представлені у вкладеній структурі, що зменшить кількість JOIN у SQL.
- MongoDB підходить для логів і транзакцій, коли потрібно швидко обробляти замовлення.
- У реальному житті дані замовлення можуть мати динамічну структуру (наприклад, замовлення різних типів можуть мати різні поля). У MongoDB можна зберігати дані у форматі JSON/BSON, де структура може змінюватися без необхідності модифікувати схему бази.

Тестування швидкодії

Операція	**MySQL(sec)**	**MongoDB(sec)**
Створення нових reviews	0.00433700	< 0.001
Знаходження всіх відгуків для конкретної книги	0.04359300	0.004
Підрахувати кількість останніх 10 замовлень зі статусом "Completed"	0.02483500	0.012
Створення нового клієнта разом із замовленням	0.03991	0.014
Отримення останніх замовлень клієнта	0.02120400	0.007

1)Створення нових reviews

MongoDB: Завдяки природній підтримці JSON-подібних документів, MongoDB дозволяє швидко вставляти складні об'єкти, такі як відгуки, з вкладеними даними.

MySQL: Для аналогічної операції необхідно взаємодіяти з кількома таблицями через INSERT з використанням зовнішніх ключів.

Доцільність: Це показує, як MongoDB справляється з масовими вставками складних даних у вигляді документів порівняно з реляційним підходом MySQL.

2)Знаходження всіх відгуків для конкретної книги

MongoDB: Підтримка індексів на рівні документів дозволяє швидко знаходити всі відгуки за book_id.

Доцільність: Це підкреслює використання індексів для пошуку даних у MongoDB, що є критично важливим для оптимізації запитів.

3)Підрахувати кількість останніх 10 замовлень зі статусом "Completed"

MongoDB: Використання aggregate дозволяє виконувати складні обчислення, такі як підрахунок загальної суми замовлень кожного клієнта, без необхідності JOIN.

MySQL: Для аналогічної операції потрібні JOIN і функції групування (GROUP BY).

Доцільність: Це демонструє, як MongoDB може обробляти складні агрегації безпосередньо на рівні колекцій, порівняно зі складнішою реляційною структурою MySQL.

4)Створення нового клієнта разом із замовленням

MongoDB: Завдяки вкладеним документам можна вставити клієнта разом із його замовленнями в один запит.

MySQL: Для цього необхідно виконати кілька INSERT-запитів у пов'язані таблиці.

Доцільність: Це ілюструє перевагу MongoDB у роботі зі складними об'єктами, тоді як MySQL вимагає дотримання реляційної моделі.

5)Отримення останніх замовлень клієнта

MongoDB: Запити зі сортуванням за датою ($sort) та обмеженням ($limit) виконуються швидко, особливо якщо індекси оптимізовані.

Доцільність: Це показує, як MongoDB справляється із задачами сортування та вибірки, які є поширеними в реальних сценаріях.

Висновки

Під час виконання лабораторної роботи було розглянуто чотири основні типи NoSQL баз даних. В результаті аналізу було обрано документноорієнтовану базу MongoDB для реалізації завдання, оскільки вона найбільше підходить для моєї предметної області (книжковий магазин) завдяки можливості зберігати вкладені структури, такі як замовлення клієнтів та відгуки.

Враховуючи предметну область, було встановлено, що деякі аспекти бізнес-логіки можна ефективніше реалізувати в NoSQL базі. Зокрема частину даних було винесено у MongoDB:

Колекції reviews для зберігання відгуків про книги.

Колекції customers з вкладеними замовленнями.

Доцільність винесення:

- Складна структура даних: MongoDB дозволяє зберігати вкладені об'єкти (наприклад, замовлення, що містять книги) без необхідності створення кількох таблиць.
- Гнучкість схеми: NoSQL база дає змогу зберігати відгуки клієнтів з різною кількістю полів без модифікації структури даних.
- Швидкість операцій: MongoDB показує високу швидкодію при масових вставках і складних агрегаціях.

Після тестування швидкодії MySQL та MongoDB було отримано такі результати за наступниими критеріями:

Масове додавання даних: MongoDB виявилася значно швидшою завдяки відсутності необхідності валідації зовнішніх ключів і гнучкій схемі.

Пошук відгуків за книгою: Запити в MongoDB з використанням індексів працюють швидше, оскільки не потребують JOIN-операцій.

Агрегація замовлень: MongoDB забезпечила вищу швидкість завдяки агрегаційному фреймворку, тоді як MySQL вимагала кількох складних JOIN і GROUP BY.