DOI: 10.32347/2412-9933.2021.47.78-82
УДК 004.6

Стецик Олексій Андрійович
Аспірант кафедри інформаційних технологій проектування та прикладної математики,
orcid.org/0000-0002-1061-0465
Київський національний університет будівництва і архітектури, Київ
Теренчук Світлана Анатоліївна
Кандидат фізико-математичних наук, доцент кафедри інформаційних технологій проектування та прикладної математики, *orcid.org/0000-0001-6527-4123*
Київський національний університет будівництва і архітектури, Київ

ПОРІВНЯЛЬНИЙ АНАЛІЗ АРХІТЕКТУР НЕРЕЛЯЦІЙНИХ БАЗ ДАНИХ

Анотація. *Статтю присвячено дослідженню проблемних питань через зростання масштабів і вимог до сучасних високонавантажених розподілених систем. Актуальність роботи забезпечується тим, що важливим компонентом кожної такої системи є база даних. У роботі висвітлено основні проблеми, що пов'язані з використанням реляційних баз даних у багатьох високонавантажених розподілених системах. При цьому основна увага спрямована на дослідження таких властивостей, як узгодженість даних, доступність і стійкість системи. Надано основні відомості про архітектуру і призначення нереляційних баз даних із широкою колонкою, баз даних за типом ключ-значення та документ орієнтованих баз даних. Показано переваги та недоліки нереляційних баз даних різних типів, які проявляються при розв'язанні різних задач залежно від призначення та особливостей системи. Обґрунтовано вибір нереляційних баз даних різних типів для порівняльного аналізу. Детально досліджені такі бази даних, як Касандра, Редіс і Монго, які тривалий час застосовуються у високонавантажених розподілених системах і вже добре зарекомендували себе серед користувачів. При цьому основна задача, яка розв'язувалася в цій статті, полягала в пошуку відповіді на питання доцільності застосування нереляційних баз даних архітектури Касандра, Редіс і Монго залежно від особливостей системи. На основі проведеного аналізу запропоновано варіанти використання цих баз даних для систем з високою кількістю запитів зчитування або запису інформації.*

Ключові слова: високонавантажена розподілена система; доступність; нереляційна база даних; узгодженість; реплікація; сегментування; стійкість

Вступ

У XXI столітті велика кількість сучасних вебзастосунків являють собою високонавантажені розподілені системи (ВНРС).

Важливим компонентом кожної такої системи є база даних. Проте через зростання масштабів і вимог до сучасних вебзастосунків змінюються обмеження та вимоги до баз даних ВНРС.

Серед основних вимог до бази даних сучасних ВНРС є здатність:

– обробляти велику кількість запитів за малий проміжок часу;

– продовжувати коректно відповідати на запити користувачів у разі виходу з ладу декількох серверів одночасно.

Проте при використанні реляційних баз даних в багатьох ВНРС виникають обмеження та ускладнення, що є наслідками проблем [1; 2]:

– які виникають при масштабуванні;

– які виникають при зміні структури таблиць;

– зниження швидкості відповіді на запит з плином часу;

– стійкості в стандартних конфігураціях.

Саме тому роботи, що спрямовані на розробку і вдосконалення баз даних для ВНРС різного призначення, лишаються актуальними.

Аналіз останніх досліджень і публікацій

Вищезазначені проблеми пов'язані з тим, що реляційні бази даних зберігають дані у вигляді таблиць списків, структура яких має бути наперед заданою, але до початку роботи програми користувачу не завжди відомо які саме колонки будуть використовуватись. До того ж, кількість потрібних колонок у таблицях може змінюватись з часом внаслідок зміни зовнішнього середовища.

Проблеми масштабування виникають, коли потрібно розділяти одну таблицю на декілька серверів. Зміна структури таблиць у реляційних базах вирішується за допомогою міграції даних,

але сама по собі міграція даних також є не простою процедурою. Окрім того, внаслідок розростання бази даних знижується швидкість відповіді на запит.

Цю проблему можна вирішувати шляхом додавання індексів, але додавання індексів збільшує об'єм бази даних, що призводить до погіршення її масштабованості.

Щодо проблеми стійкості, то згідно з теоремою Брюера, кожне розподілене сховище даних може мати тільки дві з трьох таких властивостей, як: узгодженість даних, доступність і стійкість [2].

Властивість узгодженості даних полягає в тому, що в будь-який момент часу два користувачі, які будуть робити запит до цього розподіленого сховища, отримають однакові дані.

Властивість доступності є гарантією того, що кожен запит отримає відповідь без гарантії того, що ця відповідь є останньою версією даних.

Властивість стійкості полягає в тому, що система продовжує працювати навіть якщо деякі із серверів виходять з ладу.

Згідно з [2] властивості реляційних баз даних лежать у площині узгодженості і доступності, тому реляційні бази даних не є стійкими в стандартних конфігураціях. Проте серед необхідних вимог до сучасних ВНРС все більшу роль відіграє властивість стійкості. Це призводить до стійкого зростання попиту на нереляційні бази даних, більшість з яких спроєктовані таким чином, щоб масштабуватися шляхом реплікації і сегментування даних при збільшенні об'єму роботи [3; 4].

Реплікація даних є важливим механізмом, який забезпечує стійкість нереляційних баз даних і дає змогу зберігати копії даних на різних серверах мережі [1].

Сегментування базується на розбитті даних на незалежні сегменти і застосовується для зменшення часу відповіді на запит. Для сегментування даних в нереляційних базах даних застосовують різні види стійкого хешування [5].

Додатково для досягнення швидкої відповіді на запит [4; 6]:

– в оперативній пам'яті застосовується хеш таблиці та списки з пропусками;

– в довготривалій пам'яті, окрім кешування даних, застосовуються такі структури даних, як Б-дерева та дерево із відсортованих списків.

Також в нереляційних базах даних є такі протоколи, як алгоритм пліток [7], які працюють у випадках виходу з ладу одного (декількох) серверів ВНРС і тим самим забезпечують стійкість системи.

Отже, розроблення та вдосконалення нереляційних баз даних для вебзастосунків, які працюють у режимі реального часу та для аналізу великих даних (Big Data), є актуальною задачею. Але в кожному конкретному випадку на етапі розроблення нереляційної бази даних постає питання вибору такої її архітектури, яка забезпечить максимальну продуктивність роботи ВНРС при мінімальній кількості затрачених ресурсів [4; 8].

Обґрунтування вибору найкращої архітектури бази даних, своєю чергою, потребує знань про переваги і недоліки нереляційних баз даних різних архітектур, які проявляються при розв'язанні різних задач залежно від призначення та особливостей системи.

Наприклад [4; 9]:

– урахування обмежень на структуру даних;

– урахування вимог до швидкості відновлення системи при виході з ладу одного із серверів;

– урахування вимог до узгодженості системи;

– урахування потреби забезпечення здатності системи толерувати неузгодженість на різних серверах;

– урахування кількості та характеру запитів до системи.

Мета і задачі публікації

Метою публікації є виконання порівняльного аналізу нереляційних баз даних різних архітектур. При цьому основною задачею, яка розв'язується в цій статті, є пошук відповіді на питання доцільності застосування баз даних архітектури Касандра, Редіс і Монго залежно від особливостей системи.

Виклад основного матеріалу

Наразі в сучасних ВНРС застосовуються такі типи нереляційних баз даних, як: бази даних з широкою колонкою, бази даних за типом ключ-значення та документ-орієнтовані бази даних [8].

Бази даних з широкою колонкою найбільш подібні до реляційних. Цей тип баз даних також використовує колонки, рядки і таблиці, але назва і кількість колонок може бути різною в різних рядках таблиці. Ці бази даних створені для бізнес аналітики та аналізу бізнес процесів, в яких використовуються паралельні обчислення та не використовується спільна пам'ять.

Прикладами баз даних з широкою колонкою є Касандра (Cassandra) і АчБейз (HBase).

Бази даних за типом ключ-значення беруть свій початок від бази даних Динамо компанії Амазон. На платформі цієї компанії є багато сервісів, що потребують доступу лише первинного ключа до сховища даних. Для цих операцій використання реляційної бази даних є неефективним через обмеження масштабованості та доступності [10].

Прикладами баз даних за типом ключ-значення, окрім Динамо, є Редіс та Мемкеш.

Документ-орієнтовані бази даних визнані як перехідні від простих баз даних за типом ключ-значення до складніших баз даних, які можуть зберігати пару «ключ-документ» [11]. Популярність документ-орієнтовані бази даних набули завдяки тому, що в них немає строгого обмеження на структуру, що звільняє від реалізації міграції даних.

Прикладами документ-орієнтованих баз даних є Монго (MongoDB) і Коуч (CouchDB).

Для порівняльного аналізу в цій статті вибрано нереляційні бази даних архітектур Касандра, Редіс і Монго, які вже добре зарекомендували себе серед користувачів і належать до нереляційних баз даних різних типів. Розглянемо детальніше призначення та архітектурні частини баз даних цих архітектур.

База даних Касандра розроблена компанією Фейсбук у 2008 р. для реалізації комбінації технологій розподіленого зберігання та реплікацій даних, які використовувалися в базі даних Динамо та моделі Бігтейбл від компанії Google [10].

Цей тип баз даних подібний до баз даних за типом ключ-значення за винятком того, що:

– значення у даному випадку це велика кількість колонок;

– структура колонок наперед не задана і може задаватися під час виконання програми.

Касандра має децентралізовану архітектуру, що забезпечує їй масштабованість, доступність і надійність. Ця база даних оптимізована для великої кількості операцій запису і для обробки великого потоку даних. Опишемо декілька особливостей архітектури бази даних Касандра.

Касандра використовує алгоритм стійкого хешування і віртуальні вершини для сегментування даних. Для хешування ключа ця база даних використовує алгоритм Murmur3, після чого присвоює ключі з однаковими хешами до однієї і тієї ж віртуальної вершини. При цьому одній віртуальній вершині може належати низка різних хешів. Кожному серверу з кластера присвоюється множина з віртуальних вершин таким чином, щоб навантаження на сервери кластеру було більш-менш рівномірним. Ефективним інструментом розподілу навантаження є стійке хешування при застосуванні віртуальних вершин. При видаленні або додаванні нових серверів достатньо перекинути з робочих серверів системи деяку множину віртуальних вершин і навантаження знову буде збалансоване [5].

Реплікація даних здійснюється таким чином:

– сервери одного центру даних нумеруються по колу;

– визначається фактор реплікації для запису даних на робочі сервери центру;

– кожен сервер підтримує певну множину інших працюючих серверів, застосовуючи алгоритм пліток.

База даних Монго – це нереляційна база даних за *типом ключ-значення* з відкритим кодом, яка написана на мові програмування C++ у 2009 р.

Монго являє мультиплатформову документ-орієнтовану базу даних, де документи згруповані в колекції згідно їх структури. Найважливішими рисами Монго є зносостійкість і багатопотоковість. Зносостійкість досягається за рахунок асинхронного використання реплікації «господар-слуга», тобто оновлення інформації не відбуваються негайно.

За стандартних налаштувань дані відсилаються на диск кожні 60 секунд, але цей час може бути персоналізовано. Після створення нового файлу пам'ять звільнюється. Для цього все переписується на диск. Специфіка використання пам’яті в Монго обмежує ресурс для збереження інформації про одну вершину двома гігабайтами.

Для того щоб забезпечувати високу швидкість, Монго використовує індекси подібно до реляційних баз даних. Кожен документ має ідентифікатор, за допомогою якого зроблено унікальний індекс. Крім цього індексу додаткові індексування може робити адміністратор бази даних.

Індекс може складатися з декількох полів із однієї колекції. Індекси будуються за допомогою структури Б-дерева, оскільки Б-дерево працює швидше на запит зчитування інформації, ніж на запит запису інформації. Такий вибір структури обґрунтовано тим, що господар може записувати і зчитувати файли, а слуги призначаються для резервного копіювання, а отже, із сервера слуги дозволена тільки операція зчитування.

Ідентифікація документа відбувається за допомогою унікального ключа документа [11]. Унікальним ключем може бути комбінація ідентифікатора документа та часу його створення.

База даних Редіс являє собою базу даних за типом ключ-значення. Архітектура бази даних є клієнт-серверною і складається з таких компонентів, як Редіс сервер, Редіс репліка сервера і клієнт для Редіс. Окрім того, база даних Редіс використовує оперативну пам’ять і не використовує методи керування операційної системи над віртуальною пам'яттю. Для ефективної роботи Редіс розроблено власну систему перенесення тих даних, що мало використовуються, з оперативної пам’яті на диск. Ця система віртуальної пам’яті дає змогу Редіс підтримувати множини, розмір яких є більшим, ніж оперативна пам’ять, без суттєвого зниження робочих характеристик [12].

Щодо теореми Брюера, то (таблиця):

– бази даних Редіс і Монго задовольняють умови стійкості та узгодженості;

– база даних Касандра задовольняє умови доступності та стійкості.

Щодо узгодженості Редіс, то ця база даних задовольняє цій умові, оскільки є однопотоковою.

Таблиця − ***Порівняння баз даних***

База даних	Касандра	Монго	Редіс
Мова програмування	Java	C++	C
Тип бази даних	З широким стовпчиком	Документ-орієнтована	Ключ-значення
Найкраще використовувати для:	Зберігання великої кількості даних, маючи зручний інтерфейс	Якщо потрібні робити динамічні запити	Для даних, які швидко змінюються, при обмеженні максимальної кількості даних
Наявність транзакцій	Немає	Немає	Наявні
Реплікація	Децентралізована	По типу «господар-слуга»	По типу «господар-слуга»
Площина	Доступність-стійкість	Стійкість-узгодженість	Стійкість-узгодженість

Узгодженість бази даних Монго забезпечується архітектурою «господар-слуга», оскільки господар відповідає за операцію запису нових даних і не повідомить про запис допоки дані не будуть записані в усі сервери, які є репліками цього господаря. У Касандри немає стійкої узгодженості, але узгодженість настає через деякий час після реплікації даних на всі сервери, до яких будуть надходити запити зчитування. Стійкість Касандри також можна налаштувати за рахунок доступності.

Щодо доступності, то високодоступною серед зазначених баз даних є тільки Касандра. Ця база даних досягає прийнятної доступності за рахунок децентралізованої архітектури. Якщо один із серверів виходить з ладу, то інший сервер зберігає вказівки протягом деякого часу. Ці вказівки знову передадуться серверу, який вийшов з ладу, коли відновить роботу сервер, що виходив з ладу. Редіс та Монго не мають високої доступності. Справді, коли у Монго сервер-господар виходить з ладу, центр даних не буде відповідати допоки сервер-слуга не візьме на себе його роль.

Щодо стійкості, то всі досліджені бази даних є стійкими до виходу з ладу декількох серверів.

Висновки

Проведений аналіз нереляційних баз даних, які застосовуються у високонавантажених розподілених системах, надає підстави зробити такі висновки щодо доцільності застосування баз даних архітектури Касандра, Редіс і Монго залежно від особливостей системи:

1. База даних Редіс є дуже швидкою, оскільки більшість даних зберігається в оперативній пам’яті. Проте Редіс потребує багато ручної роботи при масштабуванні, тому її доцільно використовувати в комбінації з Монго, Касандрою або іншими не реляційними базами даних.
2. Базу даних Монго доцільно використовувати в тих випадках, коли:
   − складно наперед визначити її структуру;
   − запити будуть відбуватися по декількох індексах.
3. Базу даних Касандра варто використовувати тоді, коли існує багато операцій запису, потрібна дуже велика масштабованість.
4. У майбутньому планується проаналізувати:
   − взаємодію баз даних і черг повідомлень у розподілених системах, які працюють у режимі реального часу;
   − файлові структури даних та структури даних, які працюють у оперативній пам'яті та є найбільш ефективними для роботи баз даних.

Список літератури

1. Phillips, James. (2014). Surprises in Our NoSQL Adoption Survey, blog.couchbase.com.
2. Gilbert, Seth, Lynch, Nancy A. (2012). Perspectives on the CAP theorem. *Computer*, 45, 30–36.
3. Jing, Han, Guan, Le, Jian, Du. (2011). Survey on NoSQL database. 6th International Conference on Pervasive Computing and Applications, p. 363.
4. Kleppmann, Martin. (2017). Designing Data-Intensive Applications, 79–83.
5. Karger, D., Sherman, A., Berkheimer, A., Bogstad, B., Dhanidina, R., Iwamoto, K. (1999). Web caching with consistent hashing. *Computer Networks*, 31(11-16), 1205–1206.
6. O’Neil, P., Cheng, E. (1996). The log-structured merge-tree (LSM-tree). *Acta Informatica*, 33(4), 351–385.
7. Kwiatkowska, M., Norman, G., Parker, D. (2008). Analysis of a gossip protocol in PRISM. ACM SIGMETRICS *Performance Evaluation Review*, 36(3), 17–18.

8. Tudorica, Bogdan George, Bucur, Cristian. (2011). A comparison between several NoSQL databases with comments and notes. RoEduNet International Conference 10th Edition: Networking in Education and Research, P. 1–5.

9. Karnitis, G., Arnicans, G. (2015). Migration of Relational Database to Document-Oriented Database: Structure Denormalization and Data Transformation. 27th International Conference on Computational Intelligence, Communication Systems and Networks, P. 113–115.

10. Lakshman, Avinash, Malik, Prashant. (2010). Cassandra: a decentralized structured storage system. ACM SIGOPS *Operating Systems Review*, 44, 2, 36–39.

11. Győrödi, Cornelia, Győrödi, Robert, Pecherle, George, Olah, Andrada. (2015). A comparative study: MongoDB vs. MySQL. 13th International Conference on Engineering of Modern Electric Systems (EMES), P. 1–5.

12. Baron, Cristian Andrei. (2015). NoSQL Key-Value DBs Riak and Redis. *Database Systems Journal*, 7, 7–9.

Стаття надійшла до редколегії 17.09.2021

Stetsyk Oleksii
Postgraduate student Department of Information Technology Design and Applied Mathematics,
orcid.org/0000-0002-1061-0465,
Kyiv National University of Construction and Architecture
Terenchuk Svitlana
PhD, Associate Professor Department of Information Technology Design and Applied Mathematics,
orcid.org/0000-0001-6527-4123,
Kyiv National University of Construction and Architecture

COMPARATIVE ANALYSIS OF NOSQL DATABASES ARCHITECTURE

Abstract. *This article is devoted to the study of problematic issues due to the growing scale and requirements for modern high-load distributed systems. The relevance of the work is ensured by the fact that an important component of each such system is a database. The paper highlights the main problems associated with the use of relational databases in many high-load distributed systems. The main focus is on the study of such properties as data consistency, availability, and stability of the system. Basic information about the architecture and purpose of non-relational databases with a wide column, databases of key-value type, and document-oriented databases is provided. The advantages and disadvantages of non-relational databases of different types are shown, which are manifested in solving different problems depending on the purpose and features of the system. The choice of non-relational databases of different types for comparative analysis is substantiated. Databases such as Cassandra, Redis, and Mongo, which have long been used in high-load distributed systems and have already proven themselves among users, have been studied in detail. The main task addressed in this article was to find an answer to the question of the feasibility of using non-relational databases of the architecture of Cassandra, Redis, and Mongo depending on the characteristics of the system, or record information. Based on the analysis, options for using these databases for systems with a high number of requests to read or write information are proposed.*

Keywords: high-load distributed system; availability; non-relational database; consistency; replication; segmentation; partition tolerance

---

Посилання на публікацію

APA Stetsyk, O. & Terenchuk, S. (2021). Comparative analysis of NoSQL databases architecture. *Management of Development of Complex Systems*, 47, 78–82, dx.doi.org\10.32347/2412-9933.2021.47.78-82.

ДСТУ Стецик О. А., Теренчук С. А. Порівняльний аналіз архітектур нереляційних баз даних. *Управління розвитком складних систем.* Київ, 2021. № 47. С. 78 – 82, dx.doi.org\10.32347/2412-9933.2021.47.78-82.