УДК004.65.001.63 DOI: https://doi.org/10.30837/IYF.PCEIP.2024.053

РЕПЛІКАЦІЯ ДАНИХ У РОЗПОДІЛЕНИХ ІНФОРМАЦІЙНИХ СИСТЕМАХ

Радченко О.П., Ахмедзянова О.А.
Харківський радіотехнічний фаховий коледж
Харківський комп'ютерний фаховий коледж
м. Харків, Україна
e-mail: elenaradchenko0510@gmail.com

This article is dedicated to the methods of designing and supporting databases in distributed information systems. One of the most common methods of data management in distributed systems is data replication. The article describes and discusses replication schemes, their drawbacks and advantages, as well as strategies for placing replicas in distributed systems.

Проєктування баз даних для проектів з розподіленням даних є важливим і критичним фактором розробки.

Як ключові проблеми можуть при цьому виникнути – вузли з даними можуть розміщуватися в різних географічних регіонах і в певні моменти часу може бути відмінності між станами цих даних, при цьому немає центрального органу керування, що ускладнює процес підтримки узгодженості даних; складність горизонтального масштабування вузлів баз даних.

Для зменшення цих проблем для розробників доступні певні шаблони керування. Найпоширеніші з цих шаблонів: database-per-service pattern; shared database pattern; command query responsibility segregation (CQRS) pattern; saga pattern; sharding; replication [1].

Наразі більш детально розглянемо реплікацію, як один з найпоширеніших методів керування даних в розподілених системах.

Реплікація даних – це процес створення кількох копій даних і їх зберігання в різних місцях. При цьому копії даних можна зберігати в одній системі, на локальних і зовнішніх хостах, а також на хмарних хостах.

Сучасні додатки використовують розподілену базу даних у серверній частині, де дані зберігаються та обробляються за допомогою кластера систем замість того, щоб покладатися на одну конкретну систему.

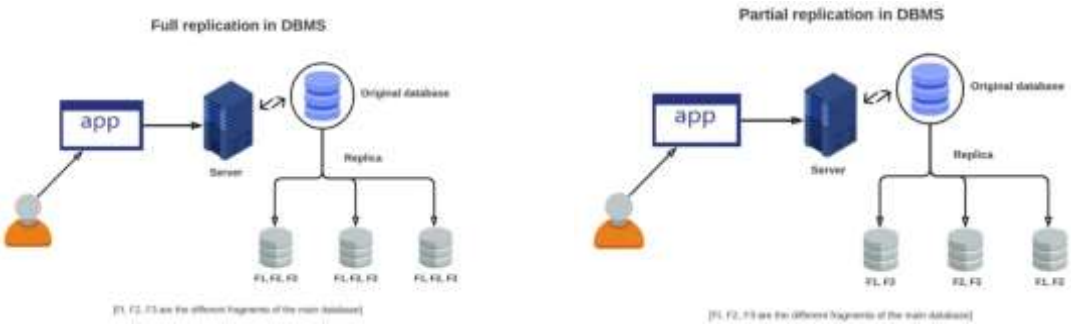
Якщо, наприклад, користувач програми бажає записати частину даних до бази даних, то ці дані будуть розділятися на певні фрагменти і при цьому кожен з цих фрагментів може зберігатися на різних вузлах розподіленої системи. Технологія бази даних також відповідає за збір і консолідацію різних фрагментів, коли користувач хоче отримати або прочитати дані.

Тиражування даних у СУБД (серверах розподілу) може здійснюватися за допомогою відповідної схеми реплікації [2].

– Повна реплікація – на кожному вузлі розподіленої системи реплікується повна база даних (рисунок 1а). У глобальній мережі дана

техніка забезпечує максимальну доступність і надмірність даних, прискорює виконання глобальних запитів. Недоліком повної реплікації є те, що процес оновлення часто відбувається повільно. Це ускладнює підтримку поточних копій даних у всіх місцях.

– Часткова реплікація – на певних вузлах реплікуються тільки певні фрагменти бази даних, які важливі для користувачів даного вузла. ні фрагменти бази даних на основі важливості даних у кожному місці (рисунок 1б). Кількість копій залежить від загальної кількості вузлів, інколи для деяких фрагментів може бути і одна копія.



а) повна реплікація б) часткова реплікація

Рисунок 1 – Схема реплікації

– Без реплікації – на кожному вузлі розміщено лише один фрагмент бази даних, і при цьому він не реплікується на інших вузлах. Перевагою такої схеми є простота відновлення (узгодження) даних, але при цьому може зменшуватися швидкість виконання запитів, оскільки кілька користувачів отримують доступ до одного сервера.

Використання реплікації має як свої переваги та і недоліки. Серед головних переваг: підвищена доступність, продуктивність, безперервність бізнесу, можливість аналітики даних без шкоди для продуктивності системи. До недоліків можна віднести: необхідність синхронізації даних на різних вузлах, витрати на обслуговування та можливе зниження продуктивності при одночасному оновленні багатьох копій [3].

Для покращення ефективності реплікації даних в останні роки достатньо багато уваги приділяють стратегіям розміщення реплік у розподілених системах.

Було запропонував динамічну стратегію розміщення даних для нових реплік, щоб знайти найкраще місце відповідно до їх термінованості. Цей метод може коригувати репліки даних, що зберігаються на кожному вузлі в гетерогенному кластері Hadoop, а також може динамічно скорочувати час відповіді додатків для великих даних [4].

Також використовують метод управління динамічною реплікацією на основі витрат, який називається CDRM. Даний метод розміщує репліки з

урахуванням ємності та ймовірності блокування граничних вузлів. Але це не охоплює погляд на обсяг даних.

Ще один з методів використовує децентралізований алгоритм розміщення копій (D-ReP) для периферійних обчислень. D-ReP базується на величині та місці попиту користувачів і ціні зберігання, щоб зменшити затримку доступу до даних. Це забезпечує найкраще покращення затримки зусиль і зниження витрат, але це не стосується завантаження крайових вузлів і гарантій продуктивності в реальному часі.

Наступний метод – динамічна та децентралізована стратегії розміщення реплік (DDRP), яка також оптимізована для багатьох цілей. Даний алгоритм може приймати рішення набагато швидше, ніж централізований, оскільки йому не потрібно виконувати складні обчислення, щоб отримати глобальне оптимальне рішення.

Таким чином, при проектуванні розподіленої системи даних необхідно враховувати початкові умови - кількість вузлів, їх географічне розміщення, необхідність тієї чи іншої інформації на різних вузлах. І вже в залежності від цих даних використовувати одну з моделей реплікації.

Список використаних джерел:

1. Designing Databases for Distributed Systems: Data Management Patterns for Microservices and Cloud-Native Applications. URL: https://dzone.com/articles/designing-databases-for-distributed-systems

2. Data Replication in Distributed Systems: The Best Guide. URL: https://hevodata.com/learn/data-replication-in-distributed-system/#intro

3. Методи масштабування реляційних баз даних: переваги, недоліки та кейси використання. URL: https://dou.ua/forums/topic/45890/

4. A dynamic decentralized strategy of replica placement on edge computing. URL:
https://journals.sagepub.com/doi/full/10.1177/15501329221115064