УДК 004.657

Лаптєва М. А., здобувачка 2 курсу спеціальності 122 Комп'ютерні науки,
Гончар В. М., асистент кафедри інформаційних технологій

РІЗНИЦЯ МІЖ SQL І NOSQL: ПЕРЕВАГИ ТА ОБМЕЖЕННЯ РЕЛЯЦІЙНИХ БАЗ ДАНИХ

Донецький національний університет імені Василя Стуса, м. Вінниця

Технологія баз даних значно розвинулася за останні десятиліття. З'явилися дві основні моделі баз даних: реляційна модель, яка використовує мову структурованих запитів (SQL), і нереляційні, або бази даних NoSQL. Обидві моделі мають різні архітектури, які надають переваги й обмеження для певних випадків зберігання та пошуку даних. Розглянемо ключові відмінності між системами баз даних SQL та NoSQL, а також висвітлимо сильні і слабкі сторони, притаманні реляційним системам керування базами даних (СКБД).

Бази даних SQL базуються на реляційній моделі, розробленій Е. Ф. Коддом у 1970 р. [1]. Дані структуруються в таблиці, що містять рядки і стовпці, з визначеними зв'язками між ними. Таблиці мають заздалегідь визначену схему, яка забезпечує сувору цілісність і стандартизацію даних. Сервери SQL надають розширені можливості управління даними, зокрема транзакції ACID (Atomicity, Consistency, Isolation, Durability), підтримку складних запитів та аналітики, а також власні програмні інтерфейси. Це забезпечує цілісність даних, але може не вистачати гнучкості та масштабованості NoSQL.

На противагу цьому, NoSQL охоплює різноманітні нетабличні та розподілені архітектури баз даних, які з'явилися на початку 2000-х рр. [2]. Найпоширеніші типи включають сховища ключ-значення, документи, графіки та сховища з широкими стовпцями. Гнучкі схеми, масштабованість між товарними серверами, висока доступність і відмовостійкість – основні переваги NoSQL. Структури даних і можливості запитів сильно відрізняються залежно від конкретної реалізації NoSQL.

Наприклад, база даних SQL моделює замовлення, використовуючи нормалізовані таблиці клієнтів, замовлень і продуктів, пов'язані ключами. Об'єднання збирають записи для транзакцій, а база даних NoSQL зберігає замовлення як окремі документи, проіндексовані за ідентифікаторами, без використання об'єднань [3]. Це полегшує ітеративне кодування, але не має жорстких схем.

Переваги реляційних баз даних:

1. Стандартизація формату даних та взаємозв'язків через фіксовані схеми таблиць сприяє цілісності, узгодженості та організації даних. Принципи нормалізації бази даних мінімізують надмірність даних.

2. Підтримка ACID-транзакцій дає змогу надійно обробляти операції створення, читання, оновлення та видалення в декількох таблицях і записах.

3. Розширена функціональність запитів та оптимізація продуктивності індексування полегшують створення складних аналітичних звітів і сценаріїв агрегації. SQL залишається основною мовою для запитів до баз даних.

4. Корпоративні можливості, як-от безпека на основі ролей, інструменти резервного копіювання / відновлення та конфігурації реплікації, забезпечують відмовостійкість і механізми контролю доступу, що є критично важливими для бізнес-додатків.

5. Фреймворки об'єктно-реляційного відображення (ORM) спрощують розробку додатків, абстрагуючи SQL-запити з допомогою інтуїтивно зрозумілих API на мовах C# та Java.

Недоліки реляційних баз даних:

1. Зміни схеми можуть містити дорогі процедури міграції даних у міру розвитку додатків.

2. Вертикальне масштабування завдяки збільшенню обчислювальної потужності одного сервера має обмеження, що обмежує економічно ефективну масштабованість СКБД.

3. Інтенсивні SQL-запити між великими таблицями можуть погано виконуватися, що впливає на чутливі до затримок додатки.

4. Технології розподіленої кластеризації SQL додають складності в розподілених транзакціях, налагодженні та балансуванні навантаження.

Розберемо на прикладі Exel реляційні бази даних. Системи RDBMS чудово справляються з цими поширеними сценаріями використання:

- структуровані бізнес-дані зі складними взаємозв'язками, що вимагають гарантій ACID;
- інтенсивні аналітичні додатки зі спеціальними потребами в запитах;
- застарілі середовища з наявними кодовими базами SQL та наборами навичок;
- системи, де цілісність даних є критично важливою, наприклад, фінансові записи або медичні дані.

NoSQL бази даних Excel:

- швидке прототипування та ітерації на гнучких моделях даних;
- надзвичайно інтенсивні завдання з простими моделями доступу до даних;
- схеми, що постійно розвиваються, наприклад, керування профілями користувачів у соціальних додатках;
- вимоги до читання / запису з низькою затримкою у високопродуктивних транзакційних додатках;
- масово розподілені архітектури з доступом до терабайт даних.

Фундаментальні конструкції даних у SQL та NoSQL суттєво відрізняються. SQL структурує дані у попередньо визначені таблиці, рядки та стовпці. NoSQL охоплює сховища документів, що організовують дані у гнучкі JSON-документи, сховища широких стовпців, що зберігають дані у стовпцях, а не в рядках, та бази даних графів, що моделюють сутності й зв'язки у вигляді вузлів / ребер мережі. Ці дуже різні парадигми даних підходять для різних архітектур додатків і моделей доступу.

Бази даних SQL дають змогу розробникам писати запити на потужній мові структурованих запитів для вставки, оновлення, видалення та пошуку записів. SQL пропонує об'єднання, агрегати, підзапити та інші конструкції для складного аналізу. Бази даних NoSQL зазвичай підтримують прості CRUD-операції та пошук із допомогою API, а не декларативних мов. Деякі винятки, як-от MongoDB, мають декларативні інтерфейси на основі JSON. Оптимізація запитів також суттєво відрізняється залежно від навігації між реляційними та нереляційними сховищами.

Понад 40 років реляційні бази даних забезпечують основу для надійного управління даними в незліченних критично важливих робочих навантаженнях. Проте повсюдне поширення неструктурованих великих даних та нові архітектури додатків вимагають додаткових нереляційних систем, оптимізованих для гнучкості, масштабованості і продуктивності. Розуміння переваг та обмежень, притаманних як SQL, так і NoSQL, допомагає приймати виважені архітектурні рішення щодо баз даних з урахуванням моделей даних, шаблонів доступу, бізнес-вимог та технічних обмежень.

Список використаних джерел

1. Codd, E. F. (1970). A Relational Model of Data for Large Shared Data Banks. Communications of the ACM, 13, 377–387.

2. Han, J., Haihong, E., Le, G., Du, J. (2011). Survey on NoSQL Database. 6th International Conference on Pervasive Computing and Applications, Port Elizabeth, 26–28 October 2011, 363–366.

3. Barata, M., Bernardino, J., Furtado, P. An overview of decision support benchmarks: TPC-DS, TPC-H and SSB. Rocha, A., Correia, A. M., Costanzo, S., Reis, L. P. (eds.). *New Contributions in Information Systems and Technologies.* Cham: Springer International Publishing, 2015. P. 619–628.