

ПОРІВНЯЛЬНИЙ АНАЛІЗ АРХІТЕКТУРНИХ ОСОБЛИВОСТЕЙ РЕЛЯЦІЙНИХ ТА NoSQL СУБД

¹Вінницький національний технічний університет

Анотація

У тезах проведено порівняльний аналіз архітектурних підходів до організації даних у реляційних (SQL) та нереляційних (NoSQL) системах управління базами даних. Розглянуто фундаментальні відмінності в моделях даних, механізмах забезпечення транзакційної цілісності (ACID проти BASE) та підходах до масштабування. Особливу увагу приділено сценаріям практичного застосування обох типів СУБД, що дозволяє обґрунтувати вибір технології залежно від вимог до проєкту. Для наочності результати систематизовано у вигляді порівняльної таблиці.

Ключові слова: реляційні бази даних, NoSQL, ACID, BASE, масштабування, аналіз.

Abstract

The thesis provides a comparative analysis of architectural approaches to data organization in relational (SQL) and non-relational (NoSQL) database management systems. Fundamental differences in data models, mechanisms for ensuring transactional integrity (ACID vs. BASE), and scaling approaches are considered. Particular attention is paid to practical application scenarios for both types of DBMS, which allows justifying the choice of technology depending on the project requirements. For clarity, the results are systematized in the form of a comparative table.

Keywords: relational databases, NoSQL, ACID, BASE, scaling, analysis.

Вступ

Вибір системи управління базами даних (СУБД) є одним із ключових архітектурних рішень при розробці сучасних програмних продуктів. Протягом десятиліть домінуючою парадигмою були реляційні бази даних, які ґрунтуються на математичній теорії відношень та підтримують мову структурованих запитів SQL [1]. Однак із зростанням обсягів даних та ускладненням веб-систем постала потреба в більш гнучких та масштабованих рішеннях, що призвело до появи класу NoSQL (Not Only SQL) баз даних. Сучасні дослідження підтверджують, що продуктивність NoSQL-систем суттєво залежить від типу виконуваних операцій та обсягів даних [2]. Метою даної роботи є порівняльний аналіз архітектурних особливостей реляційних та NoSQL СУБД для визначення раціональних сфер їхнього застосування.

Результати дослідження

У ході дослідження було проаналізовано базові принципи організації даних у двох типах систем. Реляційні СУБД, такі як PostgreSQL або MySQL, використовують попередньо визначену схему (schema) та зберігають дані в нормалізованих таблицях, пов'язаних між собою ключами. Це забезпечує високу узгодженість даних завдяки дотриманню вимог ACID (атомарність, узгодженість, ізоляція, довговічність) [1]. Натомість NoSQL бази даних пропонують гнучкі, безсхемні моделі: документо-орієнтовані (MongoDB), типу "ключ-значення" (Redis), колонкові (Cassandra) або графові (Neo4j). Вони керуються принципом BASE (базова доступність, м'який стан, кінцева узгодженість), що дозволяє досягти високої продуктивності в розподілених системах.

Ключовою архітектурною відмінністю є підхід до масштабування. Реляційні системи традиційно орієнтовані на вертикальне масштабування (нарощування потужності одного сервера), що має фізичні та економічні обмеження. NoSQL бази даних проєктуються для горизонтального масштабування - розподілу даних та навантаження на кластери з багатьох недорогих серверів, що є критичним для сучасних хмарних технологій та обробки великих даних. Практичне тестування основних операцій читання та запису на прикладі PostgreSQL, MongoDB та Cassandra демонструє, що вибір конкретної СУБД має ґрунтуватися на типі задач та вимогах до часу відповіді системи [2].

Окремо слід відзначити еволюцію самих СУБД: сучасні реляційні системи, такі як PostgreSQL, активно інтегрують підтримку JSON-документів та індексів для неструктурованих даних, що робить їх гібридними рішеннями [3]. Це дозволяє розробникам використовувати переваги обох підходів в межах однієї системи.

Для систематизації отриманих даних складено порівняльну таблицю.

Табл. 1. Порівняльний аналіз реляційних та NoSQL СУБД

Критерій	Реляційні СУБД (SQL)	NoSQL СУБД
Модель даних	Таблична, жорстка схема	Гнучка, безсхемна: документи, key-value, graph, колонки
Транзакційність	ACID (сувора узгодженість)	BASE (кінцева узгодженість)
Масштабування	Переважно вертикальне (складність горизонтального)	Горизонтальне (розподіл даних на кластері)
Продуктивність	Висока для складних запитів та JOIN-ів	Висока для простих операцій читання/запису на великих обсягах
Типові сценарії	Фінансові системи, ERP, CRM	Соціальні мережі, IoT, аналітика реального часу

Таким чином, вибір типу СУБД має базуватися на вимогах до конкретного проєкту. Якщо система потребує складної логіки, зв'язності даних та суворої узгодженості (наприклад, банківські транзакції), перевага залишається за реляційними базами даних [1]. Якщо ж пріоритетами є здатність обробляти величезні потоки напівструктурованих даних, висока доступність та гнучкість схеми (наприклад, каталоги товарів або логи подій), NoSQL є більш раціональним вибором [2]. Часто в сучасних мікросервісних архітектурах застосовується поліглотний підхід, де різні сервіси використовують різні типи баз даних залежно від своїх задач [3].

Висновки

Проведений аналіз показав, що реляційні та NoSQL СУБД базуються на принципово різних архітектурних підходах. Реляційні бази забезпечують надійність та цілісність даних за рахунок моделі ACID, що робить їх незамінними для систем із жорсткими вимогами до узгодженості [1]. Натомість NoSQL бази пропонують гнучкість, горизонтальне масштабування та високу доступність (модель BASE), що відповідає потребам сучасних Big Data-рішень. Практичне підтвердження цього положення знаходимо у порівняльному аналізі швидкодії MongoDB та MySQL на великому обсязі даних (15 ГБ): нереляційна СУБД демонструє стабільно кращі результати в усіх типах операцій - найбільший розрив фіксується при створенні таблиць (швидше на 550%) та читанні даних (швидше на 325%), тоді як на операціях модифікації перевага становить 150-250%, а найменша різниця спостерігається при видаленні таблиць (30%), що свідчить про загальну вищу ефективність NoSQL-рішень для інтенсивної обробки інформації [2]. Вибір між ними не є бінарним: архітектори програмного забезпечення дедалі частіше комбінують обидва підходи в рамках однієї системи, використовуючи переваги кожної технології для вирішення специфічних задач [3].

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Date C. J. SQL and Relational Theory: How to Write Accurate SQL Code. 3rd ed. Sebastopol, CA : O'Reilly Media, 2015. 428 p.
2. Нікітіна Т. С., Морозова О. І. Порівняльний аналіз продуктивності баз даних SQL та NoSQL. *Системи управління, навігації та зв'язку*. Полтава : ПНТУ, 2019. Вип. 1(53). С. 125–130. DOI: 10.26906/SUNZ.2019.1.125.
3. PostgreSQL: Documentation. 2025. URL: <https://www.postgresql.org/docs/> (дата звернення: 08.03.2026).

Науковий керівник: **Кожем'яко Андрій Вікторович** – к.техн.н., доцент кафедри обчислювальної техніки, Вінницький національний технічний університет, Вінниця, e-mail: kvantron@vntu.edu.ua

Сірак Владислав Олегович – студент групи ІКІ-25мс, факультет інформаційних технологій та комп'ютерної інженерії, Вінницький національний технічний університет, Вінниця, e-mail: vladsirak9@gmail.com

Supervisor: **Kozhemyako Andriy V.** – Ph.D., Associate Professor of the Computer Engineering Department,
Vinnytsia National Technical University, Vinnytsia, e-mail: kvantron@vntu.edu.ua

Sirak Vladislav O. – student of group 1KI-25ms, Faculty of Information Technologies and Computer Engineering,
Vinnytsia National Technical University, Vinnytsia, e-mail: vladsirak9@gmail.com