УДК 004.65:004.75
DOI https://doi.org/10.32782/2663-5941/2026.1.2/56

Щербина Ю.В.
https://orcid.org/0000-0003-3885-6747
Національний університет «Одеська юридична академія»

Логінова Н.І.
https://orcid.org/0000-0002-9475-6188
Національний університет «Одеська юридична академія»

Лобода Ю.Г.
https://orcid.org/0000-0001-7083-552X
Національний університет «Одеська юридична академія»

СУЧАСНІ ПАРАДИГМИ ПРОЄКТУВАННЯ БАЗ ДАНИХ ДЛЯ ХМАРНИХ ОБЧИСЛЕНЬ: ВІД РЕЛЯЦІЙНИХ МОДЕЛЕЙ ДО РОЗПОДІЛЕНИХ АРХІТЕКТУР

Стаття присвячена аналізу перспективних технологій систем керування базами даних, що використовуються для роботи з великими обсягами даних у розподілених хмарних обчислювальних середовищах. Показано, що зростання обсягів та різноманіття неструктурованих і напівструктурованих даних у сучасних інформаційних системах зумовило формування нових архітектурних парадигм NoSQL та NewSQL, які не передбачають використання жорстких структур даних і дозволяють реалізацію гнучких технік їх моделювання. Кожна з цих парадигм, протягом останніх десятиліть, була реалізована з використанням різних варіантів моделей даних, в залежності від вимог конкретного середовища експлуатації. У роботі розглянуто основні моделі NoSQL-систем, зокрема сховища типу «ключ-значення», сімейства колонок, документо-орієнтовані та графові бази даних, а також проаналізовано принципи побудови NewSQL-систем, які поєднують горизонтальне масштабування з підтримкою транзакційної семантики та SQL-інтерфейсу. Наведено теоретичні засади реляційних і NewSQL-систем, що ґрунтуються на властивостях ACID, а також NoSQL-архітектур, які проектуються з урахуванням компромісів, формалізованих теоремою CAP. Окрему увагу приділено хмарним і безсерверним архітектурам, що забезпечують еластичне горизонтальне масштабування та підвищену відмовостійкість систем зберігання даних. Сформульовано фундаментальні принципи, на які мають бути орієнтовані архітектурні та проєктні рішення масштабованих систем: простота дизайну, ефективне розподілення баз даних на окремі автономні підсистеми, слабка зв'язаність, паралелізм, економічне використання ресурсів та децентралізація. Зроблено висновок, що, попри значний прогрес у розвитку NoSQL та NewSQL-систем, залишається потреба в вдосконаленні архітектур і методів їх практичної реалізації для підтримки сучасних вимог до масштабованості, надійності та узгодженості даних.

Ключові слова: *бази даних, системи керування базами даних, сховища даних, хмарні обчислення, реляційні бази даних, розподілені бази даних, NoSQL, NewSQL, CAP, ACID, великі дані, горизонтальне масштабування, шардінг, реплікація.*

Постановка проблеми. В сучасному світі, орієнтованому на цифрові технології, бази даних (БД) становлять критичну інфраструктуру для електронної комерції, фінансових систем, промислового виробництва та соціальних платформ. Експоненційне зростання обсягів даних, потреба в обробленні інформації в реальному часі та поява нових джерел неструктурованих даних створюють безпрецедентні виклики для традиційних систем керування базами даних (СКБД).

Протягом останніх десятиріч СКБД пройшли довгий шлях еволюційного розвитку. За цей час дослідники БД та розробники програмного забезпечення дали відповідь на велику кіль-

кість викликів, що виникали по мірі розвитку відкритого інтернет-середовища і визначалися зростанням обсягів даних, розвитком технологій великих даних, хмарних обчислень та застосуванням штучного інтелекту у процесах оброблення неструктурованої інформації.

Реляційна модель «Сутність-Зв'язок» (Entity-Relationship), запропонована Едгаром Коддом у 1970 році, довгий час була надійною теоретичною основою для структурованого зберігання даних. Реляційні СКБД ефективно забезпечували транзакційну цілісність і узгодженість у централізованих системах. Водночас розвиток хмарних обчислень, соціальних мереж, Інтернету речей та розподілених застосунків виявив обмеження цього підходу: жорсткі схеми даних, складність горизонтального масштабування та низьку ефективність при обробці неструктурованої інформації.

Відповіддю на ці виклики стала поява БД типу NoSQL (Not Only SQL), які пожертвували жорсткими гарантіями узгодженості заради горизонтальної масштабованості, гнучких схем та високої доступності. Різноманітність моделей NoSQL (ключ-значення, документо-орієнтовані, колонкові, графові) дозволила адаптувати архітектури БД до специфічних вимог розподілених систем. Проте відмова від строгої узгодженості створила нові проблеми для застосунків, що потребують транзакційних гарантій.

Еволюція продовжилася з появою БД типу NewSQL, які прагнуть поєднати переваги обох підходів: масштабованість NoSQL та гарантії узгодженості реляційних систем [1, 2]. Сучасні розподілені SQL системи використовують розподілені архітектури, консенсусні протоколи та оптимізоване управління транзакціями для забезпечення ACID-властивостей у географічно розподілених кластерах.

Однак стрімкий розвиток технологій породжує нові проблеми. По-перше, відсутня єдина методологія вибору архітектури БД для конкретних сценаріїв використання, що ускладнює прийняття проектних рішень. По-друге, інтеграція технологій машинного навчання (МН) та штучного інтелекту (ШІ) вимагає нових підходів до індексування, оптимізації запитів та управління векторними даними. По-третє, поширення периферійних обчислень (Edge Computing) та багатохмарних середовищ створює виклики для забезпечення узгодженості даних при високих затримках мережі. По-четверте, зростають вимоги до економічної ефективності та енергоспоживання розподілених систем зберігання.

Таким чином, незважаючи на значний прогрес у розвитку нереляційних та гібридних моделей БД, залишається потреба в систематизації підходів до проектування архітектур СКБД для розподілених хмарних середовищ, розробці критеріїв вибору архітектур та дослідженні компромісів між узгодженістю, доступністю та продуктивністю в контексті сучасних технологічних вимог [3].

Аналіз останніх досліджень і публікацій. Постійний розвиток БД відбувається як відповідь на зростання потреб бізнесу, виробництва, а також еволюцією вимог до оброблення, зберігання та доступу до інформаційних ресурсів. Зміни у характері навантажень і масштабах даних стимулюють появу нових підходів до організації систем керування даними.

Реляційна модель даних спричинила фундаментальні трансформації у сфері керування даними, запровадивши табличну форму їх подання з формально визначеними властивостями та декларативну мову запитів SQL. Одним із її ключових досягнень стало чітке розмежування логічного рівня представлення даних і фізичного рівня їх зберігання, що усунуло необхідність реалізації навігаційного доступу до записів на прикладному рівні. Водночас результати сучасних досліджень [4, 5] свідчать про наявність принципових обмежень реляційної моделі в умовах розподілених обчислювальних середовищ, що зумовило активний пошук альтернативних архітектур. У огляді проблем масштабованості БД [5] показано, що традиційні реляційні СКБД демонструють низьку ефективність горизонтального масштабування через необхідність забезпечення строгих транзакційних гарантій у розподіленому середовищі. Це обмеження є особливо критичним для високонавантажених вебзастосунків з мільйонами користувачів, у яких вертикальне масштабування шляхом нарощування ресурсів одного сервера стає економічно недоцільним. У роботі [6] обґрунтовано, що для різних класів задач обробки великих даних доцільним є використання спеціалізованих архітектур, що сприяло формуванню та розвитку різноманітних NoSQL-підходів.

У дослідженні [7] детально проаналізовано механізми шардінгу та реплікації як ключові складові масштабованості NoSQL-систем. Шардінг забезпечує розподіл даних між незалежними вузлами без спільної пам'яті, що дає змогу досягати майже лінійного зростання продуктивності зі збільшенням кількості серверів. Реплікація, у свою чергу, підвищує доступність і надійність

системи шляхом зберігання копій даних на різних вузлах. Водночас зазначається, що застосування цих підходів породжує низку нових проблем, зокрема забезпечення узгодженості реплік, ефективне балансування навантаження між шардами та керування розподіленими транзакціями, які залишаються предметом активних наукових досліджень.

Низка наукових праць [8, 9] присвячена використанню методів машинного навчання (МН) для автоматизації процесів керування СКБД. У роботі [8] запропоновано концепцію автономних БД, у яких алгоритми МН застосовуються для оптимізації виконання запитів, автоматичного проєктування індексів і прогнозування навантаження. У дослідженні [9] продемонстровано, що в багатохмарних середовищах алгоритми ШІ ефективно розв'язують задачі динамічного масштабування, виявлення аномалій і оптимізації розміщення даних між дата-центрами, що є критично важливим для географічно розподілених систем.

Багато публікацій розглядають тенденції конвергенції різних моделей БД. Зокрема, у роботі [10] наголошується на значенні периферійних (edge) обчислень для зменшення затримок доступу до даних у розподілених системах, особливо в IoT-застосунках, де обробка інформації має здійснюватися безпосередньо поблизу джерел її генерації. У дослідженні [11] розглянуто багатомодельні БД, що підтримують роботу з різними типами даних (документними, графовими, реляційними) в межах єдиної системи, усуваючи потребу в інтеграції гетерогенних сховищ.

Незважаючи на значний прогрес у розвитку БД, існують прогалини у розумінні компромісів між різними архітектурними підходами. Зокрема бракує систематичних порівняльних досліджень продуктивності NoSQL та NewSQL у реальних сценаріях з різними типами навантаження, а також недостатньо вивчені економічні аспекти їх використання в хмарних середовищах з урахуванням витрат на обчислення, зберігання та мережний трафік. Відсутні формальні методології вибору архітектури баз даних на основі характеристик конкретних застосунків.

Сучасні NoSQL та NewSQL системи продовжують дотримуватися принципу розділення даних і програмного забезпечення. NoSQL забезпечують гнучкі схеми та високу доступність у розподілених середовищах, тоді як NewSQL поєднують реляційні моделі з ACID-гарантіями та горизонтальним масштабуванням. Шардінг, реплікація та оптимізація забезпечують ефективну обробку структурованих і напівструктурованих даних у географічно розподілених системах, що визначає актуальність даного дослідження.

Постановка завдання. Метою статті є систематизація та порівняльний аналіз існуючих архітектур БД, що призначені для роботи в розподілених хмарних обчислювальних середовищах і забезпечують можливість горизонтального масштабування, використання різних підходів до формального опису інформаційних об'єктів та реплікації даних для підвищення надійності їх зберігання.

Виклад основного матеріалу

1. Проблеми розподілених обчислювальних хмарних середовищ

Реалізація реляційних СКБД вимагає значних витрат на інфраструктуру, розраховану на пікові навантаження. Ця проблема спонукає ринок БД до пошуку більш гнучких рішень, з метою оптимізації технологічних інвестицій.

При порівняльному дослідженні реляційних та NoSQL систем виявлено, що традиційні БД демонструють коефіцієнт використання ресурсів 15-30% у нормальному режимі, що призводить до значних операційних витрат. Вертикальне масштабування (scale-up) потребує дорогого high-end обладнання і має фізичні межі розширення, тоді як горизонтальне масштабування (scale-out) реляційних БД залишається складним через необхідність підтримки ACID-гарантій у розподіленому середовищі [12].

Перехід до хмарних рішень дозволив забезпечити економію до 40-60% порівняно з on-premise рішеннями за рахунок еластичності та відсутності витрат на утримання власної інфраструктури. Це особливо критично для застосунків з непередбачуваним або циклічним навантаженням [10].

Сучасні застосунки генерують гетерогенні дані: структуровані транзакції, напівструктуровані JSON-документи, графи соціальних зв'язків, часові ряди IoT-сенсорів. Виникає необхідність у багатомодельних БД, що підтримують різні типи даних в єдиній системі, усуваючи складність інтеграції спеціалізованих сховищ та дублювання даних.

Поширення IoT та мобільних застосунків актуалізує проблему периферійних обчислень – обробки даних поблизу джерел їх генерування. Традиційна централізована архітектура створює неприйнятні затримки (100-500 мс) для застосунків реального часу типу автономного транспорту чи промислової автоматики. Хмарно-орієнтовані БД розміщують репліки даних географічно близько до користувачів, забезпечуючи латент-

ність менш 10 мс, але ускладнюють підтримку глобальної узгодженості.

Формально, хмарні обчислення можуть бути виражені моделлю, яка описує процедури надання користувачам доступу до загальних обчислювальних ресурсів, збоку постачальників мережних послуг. Головними характеристиками такої моделі є:

– самообслуговування на вимогу, коли надання ресурсів здійснюється без безпосередньої взаємодії з провайдером;

– об'єднання ресурсів, розмір, місце розташування і структура яких, приховані від користувача;

– швидка еластичність, коли процес доступу до інформаційних ресурсів відбувається швидко і у користувача виникає ілюзія необмеженої масштабованості;

– контрольна послуга, коли продуктивність та використання ресурсів контролюється, вимірюється та оптимізується автоматично.

По мірі розширення масштабів хмарних обчислень, на передній план виходять проблеми інтернет-сервісів, які передбачають роботу з великими обсягами даними. Такі компанії як, наприклад, Amazon, Meta та Google утворюють одне з найбільших у світі розподілених сховищ і платформ оброблення даних. Для обслуговування трафіку, що забезпечує їх роботу, необхідні значні обчислювальні ресурси. Проблема може бути вирішена двома способами. Перший – збільшенням потужності окремих вузлів (вертикальне масштабування, scale-up), а другий – збільшення кількості недорогих обчислювальних машин, об'єднаних в одному кластері (горизонтальне масштабування, scale-out). Надійну роботу такого кластеру можна забезпечити, побудувавши його так, щоб він продовжував працювати навіть у разі відмови окремих його складових.

Для традиційних реляційних СКБД історично підходив переважно перший спосіб, але він є достатньо витратний, і, тому, набувають популярності технології керування, що орієнтовані на другий спосіб, який дозволяє реалізувати додавання серверів по мірі зростання об'ємів даних та кількості користувачів, що до них звертаються. Їх практична реалізація відбувається на технологічному рівні СКБД, що працюють у хмарних середовищах [13].

Технології, побудовані на принципі автоматизованого керування багатомашинними кластерами з боку хмарного провайдера, називають безсерверними (serverless). Це означає, не відсутність сервера взагалі, а те, що керування інфраструктурою та масштабуванням приховане від користувача. Бази даних такого типу пропонують модель, за якої управління інфраструктурою здійснюється хмарним провайдером, що дозволяє розробникам зосередитися на створенні прикладних програм [14].

Таким чином, на поточному етапі розвитку БД основними проблемами, на розв'язанні яких зосереджують свої зусилля фахівці, є побудова ефективних безсерверних хмарних моделей розгортання БД, здатних працювати з великими обсягами даних і, в той же час, забезпечувати їм необхідний рівень надійності та захисту.

2. Теоретичні засади та принципи побудови масштабованих систем

Хоча еволюційний шлях розвитку СКБД можна простежити починаючи з 1960-го року, але, саме, широке застосування реляційної моделі після 1980 року, стало початком справжньої революції в керуванні даними. Її простота, зрозумілість та ефективність обумовили причину великої дослідницької активності та виникнення вкрай успішних програмних реалізацій СКБД цього типу. Такі реляційні СКБД висувають на перший план завдання забезпечення узгодженості даних та підтримки концепції транзакцій. Практична реалізація цих завдань ґрунтується на властивостях ACID (Atomicity, Consistency, Isolation, Durability), які забезпечують атомарність виконання транзакцій, узгодженість станів бази даних, ізоляцію паралельних операцій та стійкість збереження результатів у разі збоїв.

Модель ACID була фундаментом, на якому десятиліттями будувались реляційні СКБД. Водночас потреба у масштабованій обробці великих обсягів даних у розподілених хмарних середовищах зумовила розвиток нереляційних систем типу NoSQL, архітектури яких описуються через теорему CAP (Consistency, Availability, Partition tolerance) [15], описує фундаментальні компроміси між узгодженістю, доступністю та стійкістю до поділу в розподілених системах. Узгодженість передбачає спостереження клієнтами єдиного актуального стану даних, доступність гарантує отримання відповіді на кожний запит навіть за наявності відмов окремих вузлів, а стійкість до поділу визначає здатність системи коректно функціонувати в умовах порушення зв'язності мережі.

Теорема CAP стверджує, що з трьох можливих комбінацій вимог в поточний момент доступні тільки дві. Таким чином, архітектури NoSQL-систем можуть, наприклад, забезпечувати стійкість до поділу та узгодженість або стійкість до поділу та доступність, але забезпечення всіх трьох

властивостей одночасно є неможливим у розподіленому середовищі.

Як інженерна відповідь на ці обмеження було сформульовано концепцію BASE (Basically Available, Soft state and Eventual consistency) [16], яка описує клас систем, що орієнтуються на високу доступність (Availability) і стійкість до поділу (Partition tolerance) за рахунок послаблення вимог до негайної узгодженості (Consistency). BASE-системи залишаються доступними навіть у разі збоїв, допускають тимчасову неузгодженість і гарантують досягнення кінцевої узгодженості після стабілізації мережі.

Такий підхід дозволив NoSQL-системам досягти високої масштабованості в географічно розподілених середовищах, де мережеві затримки та поділи сегментів є неминучими. Провідні NoSQL-платформи, зокрема Apache Cassandra, Amazon DynamoDB, MongoDB та Riak, реалізують принципи BASE для забезпечення високої доступності та безперервності сервісів навіть у разі відмов окремих вузлів або цілих дата-центрів.

Матеріальним фундаментом NoSQL-архітектур є розподілені горизонтально масштабовані системи, що використовують технології кешування, шардінгу та реплікації. Кешування забезпечує зменшення затримок доступу до даних за рахунок зберігання часто використовуваної інформації в оперативній пам'яті. Шардінг реалізує горизонтальне масштабування за рахунок розподілу даних на окремі фрагменти, однак супроводжується викликами міжвузлової взаємодії, балансування навантаження та забезпечення відмовостійкості. Реплікація передбачає зберігання копій даних на різних вузлах системи, підвищуючи надійність і доступність, проте потребує вирішення задачі узгодженості реплік, особливо за умов асинхронного оновлення [17].

Таким чином, сучасні NoSQL системи будуються на основі розподілених, горизонтально масштабованих обчислювальних платформ, які забезпечують зростання обсягів збережених даних та підвищення надійності за рахунок реплікації. У хмарних середовищах горизонтальне масштабування реалізується через динамічне керування екземплярами БД і автоматичне масштабування відповідно до поточного трафіку.

Архітектурні та проєктні рішення масштабованих систем повинні бути орієнтованими на наступні фундаментальні принципи:

– простота дизайну, що полегшує розроблення, розгортання та супровід системи;

– ефективне розділення на підсистеми, кожна з яких здатна виконувати свої функції автономно;

– слабка зв'язаність, яка дозволяє забезпечувати гнучкість у виборі стратегій оптимізації складових підсистем;

– паралелізм, який дозволяє одночасне виконання кількох складових завдань, користуючись загальними ресурсами;

– економне використання ресурсів з ціллю мінімізації обчислювальних ресурсів;

– децентралізація, яка передбачає розміщення підсистем на окремих серверах і забезпечує високу масштабованість та доступність.

Із сказаного витікає, що проектування високопродуктивної масштабованої системи вимагає комплексного підходу, який враховує як функціональні, так і нефункціональні вимоги до СКБД. Для цього мають бути застосовані перевірені принципи і шаблони проєктування, що забезпечують гнучкість системи, її надійність та здатність адаптуватись до поточних умов експлуатації.

3. Моделі нереляційних баз даних типу NoSQL

Сьогодні, в області хмарних інформаційних технологій, термін NoSQL означає широкий клас СКБД, які не передбачають обов'язкового використання таблиць і не користуються мовою SQL для керування даними. Постачальники вебпослуг висувають до розробників вимогу на створення технології зберігання і маніпулювання даними, що умовно може бути вираженою як «один розмір підходить усім». Таке неформальне завдання привело до виникнення нового альтернативного підходу, який дозволяє створювати різні варіанти нереляційних БД із гнучкою схемою і різними моделями представлення інформаційних об'єктів [18].

Термін NoSQL застосовується до сімейства БД, архітектури яких орієнтовані на розв'язання проблем масштабованості та доступності в розподілених середовищах шляхом ослаблення вимог до строгої узгодженості або транзакційної атомарності. На відміну від реляційної моделі, такі системи використовують агрегатний підхід до організації даних. Агрегат – це колекція пов'язаних об'єктів, які розглядаються як єдине ціле при виконанні операцій читання та запису, що спрощує розподілене керування узгодженістю даних.

Крім вирішення проблем, пов'язаних з горизонтальним масштабуванням, системи NoSQL забезпечують: гнучке представлення даних без фіксованих схем, що дозволяє змінювати структуру записів у процесі їх еволюції; скорочення

часу розроблення за рахунок спрощених моделей доступу до даних; високу продуктивність при роботі з великими обсягами інформації; можливість створення еластичних застосунків, що здатні справлятися з різкими сплесками навантаження.

Виділяють чотири основні класи NoSQL-систем: сховища типу "Ключ-значення" (Key-Value stores), сховища "Сімейства колонок" (Column-family stores), "Сховища документів" (Document stores) та "Графові бази даних" (Graph databases). Окрім них існують спеціалізовані системи з подібними властивостями масштабованості та гнучкості, однак ці чотири класи становлять основу сучасних NoSQL-архітектур.

В основу БД типу "Ключ-значення" закладені асоціативні масиви у яких зберігаються пари ключів та значень, що їм відповідають. Кожен ключ має бути простим унікальним об'єктом, а відповідне до нього значення, може включати списки, множини або хеш-значення, які дозволяють створювати більш складні кортежі даних. Для маніпулювання даними в таких базах виконуються операції додавання нових пар ключ-значення, їх пошук через значення ключа та видалення окремих пар. Модель цього типу дозволяє швидко і ефективно додавати об'єкти нових типів, але її застосування знижує гнучкість процедур виконання запитів. Прикладом, баз даних, що використовують цей тип моделі є Dynamo від Amazon, недоліком якої є обмежений інтерфейс для виконання запитів.

Сховища даних типу "Сімейства колонок", по суті, є розширеним варіантом моделі "Ключ-значення" з двовимірним значенням ключа, яке складається зі значення ключа стовпця і ключа рядка. В окремих випадках можливі додаткові розширення ключа, які називають просторами ключів або доменами. Так само як і у сховищі типу "Ключ-значення", величини, що зберігаються у кортежах, не інтерпретується системою далі. У сховищах даних типу "Сімейства колонок" ключі можуть мати безліч вимірювань і це призводить до структури, аналогічної багатовимірному асоціативному масиву.

У базах даних NoSQL типу, які входять у групу, що об'єднуються назвою "Сховища документів", дані зберігаються у певному структурованому форматі. Незважаючи на те, що такий формат є фіксованим, він, у той же час, залишається достатньо гнучким. Наприклад, у сховищах, побудованих на основі на основі JSON (JavaScript Object Notation), можуть зберігатись документи з різними наборами атрибутів. На абстрактному рівні "Сховища документів" нагадують БД типу "Ключ-значення". Вони також містять значення, які програма може читати або видаляти, за допомогою ключа. Такі документо-орієнтовані БД автоматично генерують новий ключ, у разі створення нового документа.

Зазвичай, у документо-орієнтованих БД, документи зберігаються в колекціях. У порівнянні з реляційними БД, еквівалентом колекції є запис (кортеж), а документа – відношення (таблиця). У складі документів можуть зберігатись різні набори атрибутів, що пов'язані з форматом файлу, яким можна маніпулювати за допомогою відповідної мови програмування, проте, тут не можливе використання моделі «сутність-зв'язок».

Дані у документо-орієнтованих базах зберігаються у вигляді пар ключ-значення. Але, не зважаючи на це, в них можна користуватись для виконання запитів не тільки значенням ключа, а й значеннями атрибутів, таких, як FirstNm, LastNm, та інших.

БД типу "Сховищі документів" уявляють собою реалізацію ефективного підходу до моделювання даних з урахуванням конкретних вимог, що визначаються середовищем їх експлуатації, але вони відрізняються дещо нижчою продуктивністю, у порівнянні зі сховищами даних типу "Ключ-значення". До їх складу можна віднести такі відомі бази як Riak, MongoDB та CouchDB.

У "Графових базах даних", як це видно із назви, дані представлені у вигляді графів. БД цього типу найкраще підходять для подання даних з великою і гнучкою кількістю взаємозв'язків. Це особливо зручно коли інформація про такі взаємні зв'язки так само важлива, як і самі представлені дані. Прикладом таких даних можуть бути соціальні відносини або географічні дані. Графова модель БД дозволяє виконувати запити, використовуючи графову структуру. Наприклад, використовуючи взаємні зв'язки між вузлами або найкоротші шляхи. Реалізації графових БД дозволяють ефективно підтримувати такі запити, використовуючи добре відомі графові алгоритми [19].

Отже, різні моделі NoSQL-систем відрізняються способами організації даних, рівнем масштабованості й гнучкістю запитів. Кожна модель оптимізована під певний клас задач, тому вибір конкретної NoSQL-архітектури повинен базуватися на характері даних, вимогах до узгодженості та типах запитів, які повинна підтримувати система.

4. Модель реляційної бази даних типу NewSQL

NewSQL уявляє собою клас систем керування реляційними БД нового покоління, орієнтованих

на онлайн-обробку транзакцій (Online Transaction Processing – OLTP), які поєднують горизонтальну масштабованість розподілених архітектур із збереженням семантики ACID [20]. На відміну від класичних реляційних СКБД, NewSQL-системи спроєктовані для роботи в кластерних і хмарних середовищах без відмови від транзакційної узгодженості.

Системи цього класу долають обмеження традиційних реляційних СКБД за рахунок використання розподілених архітектур, керування розміщенням даних і сучасних механізмів синхронізації транзакцій. Окремі реалізації також застосовують колонкове зберігання та обробку в оперативній пам'яті (in-memory processing) для підвищення продуктивності, а також використовують багатопроцесорні обчислювальні платформи.

Модель NewSQL, на відміну від звичайної реляційної моделі, підтримує горизонтальне масштабування разом зі збереженням ACID-властивостей. Це забезпечує можливість виконання транзакцій у географічно розподілених хмарних середовищах під час високих інформаційних навантаженнях.

Незважаючи на відмінності у внутрішніх архітектурах, усі NewSQL-системи підтримують реляційну модель даних і використовують SQL як інтерфейс доступу. Практичні реалізації таких БД, зазвичай, використовують архітектури мікросервісів з виділеними компонентами для обробки запитів, координації транзакцій та керування зберіганням даних. Для використання у хмарних середовищах такі NewSQL бази даних, як, наприклад, Amazon Aurora та Google Cloud Spanner, забезпечують еластичну масштабованість за рахунок розподілу ресурсів зберігання та обчислень, дозволяючи незалежно масштабувати кожен компонент залежно від навантаження.

Забезпечення безпеки у таких розподілених середовищах є складним завданням, тому сучасні NewSQL системи впроваджують багаторівневі механізми захисту, включаючи автентифікацію, керування правами доступу та криптографічне шифрування даних [21].

Висновки. Із наведеного аналізу випливає, що починаючи з початку XXI століття проєктування СКБД супроводжується переходом до нових архітектурних парадигм, зумовлених потребою обробки великих обсягів даних у динамічних розподілених середовищах. Незважаючи на те, що реляційна модель даних і надалі залишається домінуючою в СКБД, традиційні централізовані архітектури не забезпечують достатньої гнучкості, доступності та можливості горизонтального масштабування.

У відповідь на ці обмеження були сформовані напрями NoSQL та NewSQL, які пропонують різні підходи до організації, доступу та еволюції даних. Системи NoSQL орієнтовані на гнучке подання неструктурованих і напівструктурованих даних, високу доступність і масштабованість у розподілених середовищах, тоді як NewSQL-архітектури поєднують ці властивості зі збереженням транзакційної семантики та гарантій цілісності реляційних даних.

Подальше поширення хмарних і периферійних обчислень сприяє переходу до безперервного керування даними в географічно розподілених середовищах. У цих умовах спостерігається тенденція до інтеграції засобів машинного навчання та штучного інтелекту у СКБД, а також до розвитку інтелектуальних інтерфейсів, що приховують внутрішню складність систем і водночас підвищують їхню ефективність та надійність.

Водночас аналіз показує, що сучасні архітектури NoSQL та NewSQL залишаються різноманітними і недостатньо стандартизованими, що ускладнює їх порівняльну оцінку та практичне впровадження. Це зумовлено швидкими змінами у технологічному середовищі та необхідністю адаптації СКБД до нових вимог щодо продуктивності, масштабованості та надійності.

Список літератури:

1. Pavlo J., Aslett A. What's Really New with NewSQL? *ACM SIGMOD Record*, Volume 45, Issue 2. Pages 45–55. DOI: https://doi.org/10.1145/3003665.3003674

2. Wisal K., Teerath K., Zhang C., Kislay R., Arunabha M. R., Bin L. SQL and NoSQL Database Software Architecture Performance Analysis and Assessments – A Systematic Literature Review. *Big Data and Cognitive Computing*. 2023, 7, 97. DOI: https://doi.org/10.3390/bdcc7020097

3. Naresh Kumar Miryala. Emerging Trends and Challenges in Modern Database Technologies: A Comprehensive Analysis. *International Journal of Science and Research* (IJSR), Volume 13 Issue 11, November 2024. Pages: 1686–1696. DOI: https://dx.doi.org/10.21275/MS241126103744

4. Umit Mester. How Database Development Has Evolved Over the Years in the Web Development Area? *International Journal of Computer Trends and Technology* (IJCTT), vol. 73, no. 3, pp. 103–111. 2025. URL: https://doi.org/10.14445/22312803/ IJCTT-V73I3P113

5. Cattell R. Scalable SQL and NoSQL Data Stores. *ACM SIGMOD Record*, Vol. 39, Iss. 4, pp. 12–27. DOI: https://doi.org/10.1145/1978915.1978919

6. Onwuama T.U, Odii J.N, Nwokoma F.O, Okpalla C.L Current computing trend in Database Design. *IEEE 3rd International Conference on Electro-Technology for National Development* (NIGERCON). March 2023. pp. 373–382. URL: https://www.researchgate.net/publication/369369629_Current_computing_trend_in_Database_Design

7. Aniko V´agner, Mustafa Al-Zaidi. Sharding and Master-Slave Replication of NoSQL Databases: Comparison of MongoDB and Redis. *In Proceedings of the 12th International Conference on Data Science, Technology and Applications* (DATA 2023), pages 576–582. DOI: https://doi.org/10.5220/0012142700003541

8. Lieponiene J. Recent Trends in Database Technology. Baltic J. *Modern Computing*, Vol. 8 (2020), No. 4, pp. 551–559. https://doi.org/10.22364/bjmc.2020.8.4.06

9. Unuriode O. A., Durojaiye, M. O., Yusuf, Y. B., Okunade, O. L. The Integration of Artificial Intelligence in to Database Systems (AI-DB Integration Review). *International Journal on Cybernetics & Informatics* (IJCI) Vol.12, No.6, December 2023. pp. 161–172. DOI: https://doi.org/10.5121/ijci.2023.1206012

10. Aparna S. Bhande, Nidhi J. Sharma, Sujata V. Barabde. Emerging Trends in Database Design: Toward Improved Modeling and Development Practices*. The European Physical Journal Conferences*. June 2025. DOI: https://doi.org/10.1051/epjconf/202532801065

11. Toorpu A. R. Unified Data Access in Multi-Model Databases: Querying Relational, Graph, and Document Data Seamlessly. *IJGIS.* vol. 2. no. 3. May 2025. DOI: https://doi.org/10.63412/4qcgbv22.

12. Salim Al Maamari, Mohammad Nasar. A Comparative Analysis of NoSQL and SQL Databases: Performance, Consistency, and Suitability for Modern Applications with a Focus on IoT. 2025. *East Journal of Computer Science* 1(2). 10–15. DOI: https://doi.org/10.63496/ejcs.Vol1.Iss2.76

13. Siva Prasad Nandi. Serverless databases: Architectural evolution, implementation strategies, and future directions in cloud-native data management. *World Journal of Advanced Research and Reviews*, 2025, 26(02), pp. 567–574. DOI: https://doi.org/10.30574/wjarr.2025.26.2.1591

14. Sai Venkata Kondapalli. Serverless Database Solutions: The Next Evolution in Cloud Data Management. *European Journal of Computer Science and Information Technology*. 2025. 13(15). pp. 110–118. DOI: https://doi.org/10.37745/ejcsit.2013/vol13n15110118

15. Pradeep Bhosale. Data Consistency Models in Distributed Systems: CAP Theorem Revisited. *International Journal on Science and Technology* (IJSAT). Vol. 14. Iss. 3. July-September 2023. URL: https://www.ijsat.org/papers/2023/3/1408.pdf

16. Badwaik N. Designing System Architecture with High Availability and Scalability. *American Research Journal of Computer Science and Information Technology.* Vol. 7. Iss. 1. pp. 6–10. DOI: https://doi.org/10.21694/2572-2921.24002

17. Sydney Anuyah, Emmanuel Bolade, Oluwatosin Agbaakin. Understanding Graph Databases: A Comprehensive Tutorial and Survey. Published in arXiv.org 15 November 2024. DOI: https://doi.org/10.48550/arXiv.2411.09999

18. Ginard S. Guaki. A Comparative Analysis of Relational, NoSQL and NewSQL *Database System*. 2024. DOI: https://doi.org/10.13140/RG.2.2.27904.03849

19. Bhupender Kumar Panwar. NoSQL and NewSQL Databases: Scaling beyond relational limits: Aarchitectures, performance and future directions. *World Journal of Advanced Engineering Technology and Science*s. 2025. 15(01). 135–142. DOI: https://doi.org/10.30574/wjaets.2025.15.1.0193

Shcherbyna Yu.V., Loginova N.I., Loboda Yu.G. MODERN PARADIGMS OF DATABASE DESIGN FOR CLOUD COMPUTING: FROM RELATIONAL MODELS TO DISTRIBUTED ARCHITECTURES

The article is devoted to the analysis of promising technologies of database management systems used for working with large volumes of data in distributed cloud computing environments. It is shown that the growth in volume and diversity of unstructured and semi-structured data in modern information systems has led to the formation of new architectural paradigms NoSQL and NewSQL, which do not involve the use of rigid data structures and allow the implementation of flexible data modeling techniques. Each of these paradigms, over recent decades, have been implemented using various variants of data models, depending on the requirements of specific operating environments. The paper examines the main models of NoSQL systems, particularly key-value stores, column families, document-oriented and graph databases, and analyzes the principles of building NewSQL systems, which combine horizontal scaling with support for transactional semantics and SQL interface. The theoretical foundations of relational and NewSQL systems, based on ACID properties, as well as NoSQL architectures, which are designed considering trade-offs formalized by the CAP theorem,

are presented. Special attention is paid to cloud and serverless architectures that provide elastic horizontal scaling and enhanced fault tolerance of data storage systems. Fundamental principles are formulated that should guide architectural and design decisions for scalable systems: design simplicity, efficient distribution of databases into separate autonomous subsystems, loose coupling, parallelism, economical use of resources, and decentralization. It is concluded that, despite significant progress in the development of NoSQL and NewSQL systems, there remains a need for improving architecture and methods of their practical implementation to support modern requirements for scalability, reliability, and data consistency.

Keywords: *databases, database management systems, cloud computing, distributed databases, relational databases, NoSQL, NewSQL, ACID, CAP, big data, horizontal scalability, sharding, replication.*

Дата першого надходження статті до видання: 21.01.2026
Дата прийняття статті до друку після рецензування: 13.02.2026
Дата публікації (оприлюднення) статті: 08.04.2026