**УДК: 004.65:004.43**
**JEL Classification: O 31, O32**
**doi: 10.31767/nasoa.1-2-2022.09**

***О. Л. ЄРШОВА,***
*кандидат економічних наук, доцент,*
*старший науковий співробітник лабораторії дистанційного навчання,*
*Інститут професійної освіти НАПН України;*
*e-mail: midnight2003@nasoa.edu.ua,*
*ORCID: 0000-0002-3801-9730;*
***О. В. СТАВИЦЬКИЙ,***
*кандидат економічних наук, доцент,*
*доцент кафедри економіко-математичних*
*дисциплін та інформаційних технологій;*
*Національна академія статистики, обліку та аудиту;*
*e-mail: OVStavitskiy@nasoa.edu.ua,*
*ORCID: 0000-0002-2114-0892*

# Дослідження сучасних тенденцій у технологіях баз і сховищ даних як технологічної та архітектурної основи для створення програмних і інтелектуальних систем засобами сучасних мов програмування. Частина 2

*Стаття містить аналітичний огляд тенденцій розвитку технології баз даних, зроблений на базі звітів, підготовлених за результатами восьми зустрічей фахівців у галузі баз даних, які відбулись у період 1988–2013 років. Об'єктами аналізу є найцікавіші прогнози, що містяться у звітах, а саме їх реалістичність, точність, прагматизм або, навпаки, утопічність чи кон'юнктурність.*

*Стаття складається з двох частин.*

*В першій частині розглядаються і оцінюються прогнози, висунуті у звітах про перші чотири зустрічі, що відбулися в 1988, 1990, 1995, 1996 роках. Ці прогнози стосуються створення, розвитку і застосування систем підтримки прийняття рішень, програмно-апаратних комплексів для управління базами даних, графічних процесорів, операційних систем, інтерфейсу мови структурованих запитів, додатків баз даних, розподілу інформації, універсальних систем управління базами даних, критеріїв оптимізації запитів, інтелектуальний аналізу даних усередині системи управління базами даних. Докладно розглянуто тематику досліджень у галузі баз даних, яка набула пріоритетності в той період: облік і розрахунки даних, безпека та конфіденційність даних, реплікація та узгодження даних, забезпечення структурованості даних, інтелектуальний аналіз даних, сховища даних.*

*Друга частина присвячена огляду прогнозів, що містяться у звітах про зустрічі, які відбулися у 1998, 2003, 2008 та 2013 роках. Прогнози стосуються самоналаштування систем баз даних, переосмислення традиційної архітектури систем баз даних завдяки новим апаратним можливостям. Окрема увага в них приділена можливості роботи зі структурованими та неструктурованими даними всередині архітектури системи управління базами даних, підтримці технології великих даних, окреслено проблематику досліджень для реалізації її потенціалу.*

***Ключові слова:*** *бази даних, система управління базами даних, сховища даних, система підтримки прийняття рішень, аналітичне оброблення даних, мова структурованих запитів, інтелектуальний аналіз даних, транзакційні запити, аналітичні запити.*

***O. YERSHOVA,***
*PhD (Economics), Assotiate Professor,*
*Senior Researcher at the Distance Learning Laboratory,*
*Institute of Vocational Education NAES of Ukraine;*
***O. STAVYTSKYI,***
*PhD (Economics), Assotiate Professor,*
*Associate Professor of Department*
*for Econometric Disciplines and Information Technologies*
*National Academy of Statistics, Accounting and Audit*

# A Study of Modern Trends in Database and Data Repository Technologies as the Technological and Architectural Basis for the Creation of Software and Intelligent Systems by Means of Modern Programming Languages. Part 2

*The article contains an analytical review of developments in database technologies, made on the basis of reports prepared by the results of eight meetings of database specialists held throughout 1988–2013. Objects of the analysis are most interesting predictions given in the reports: their realism, accuracy, pragmatism or, vice versa, utopianism or opportunism.*

*The article consists of two parts.*

*Part 1 is devoted to analysis and evaluation of predictions made in the reports of the four earlier meetings held in 1988, 1990, 1995, and 1996. These predictions are about creation, development and uses of decision support systems, database appliances, graphic processing units, operating systems, interface for structured query language, database applications, information distribution, universal database management systems, query optimization criteria, intellectual analysis of database within database management systems. A detailed description of research themes in the field of databases, which got the priority status in that time, is given: recording and computation of data, security and confidentiality of data, replication and harmonization of data, structuring of data, intellectual analysis of data, data warehouses.*

*Part 2 is devoted to an analytical review of the predictions contained in the reports on the meetings held in 1998, 2003, 2008, and 2013. The predictions are about self-adjustment of database systems, rethinking of the traditional database architecture as a result of new hardware capabilities. They make special emphasis on the feasibility of manipulations with structured and unstructured data within DSS architecture, support of Big Data technology, with outlining the themes of research aimed at implementation of its potential.*

**Keywords:** *databases, database management system, data warehouses, decision support systems, analytical data processing, structural query languange, data mining, transactional queries, analytic queries.*

**5. *Зустріч у Асіломарі 1998 року.***

*Системи управління базами даних у стилі "plug and play"*. У зв'язку зі зростанням відносної вартості людського фактора в комп'ютерних системах потрібно, щоб майбутні комп'ютерні системи стали повністю автоматичними: автоінстальованими, автокерованими, авторемонтованими та автопрограмованими. У тому числі потрібно забезпечення самонастроювання систем баз даних, тобто видалення маси параметрів налаштування продуктивності, які мають визначати користувачі у поточних продуктах. Зокрема, мають з'явитись методи автоматичного вибору індексів.

Важливість проблеми самонастроювання систем баз даних у XXI столітті усвідомлюється всіма великими вендорами SQL-орієнтованих СУБД. Очевидно, першою компанією, у якій почалися масштабні дослідження у цьому напрямі, була "Microsoft" (проєкт AutoAdmin під керівництвом Сураджита Чаудхарі розпочався 1996 р. [6, 7]).

На початку цього проєкту було розроблено засіб Index Tuning Wizard, який аналізував поточне робоче навантаження СУБД (операції вибірки даних, вставки, модифікації та видалення рядків таблиць) та встановлювало, за якого набору індексів на відповідних

таблицях набір операцій міг би оброблятися СУБД найбільш ефективно. При аналізі робочого навантаження вже використовувалися методи data mining. Надалі у цьому засобі почали враховуватися як індекси, а й матеріалізовані уявлення та інші об'єкти бази даних, наявність чи відсутність яких впливає на ефективність оброблення операцій.

Засоби самоналаштування, наявні сьогодні в передових СУБД, здебільшого спрямовані на те, щоб допомогти адміністраторам баз даних налаштувати фізичну схему бази даних, щоб вона найкраще відповідала особливостям поточного середовища використання. Крім того, в системи широко впроваджуються засоби моніторингу, результати яких дозволяють адміністраторам розпізнати як власні недоліки при налаштуванні бази даних, так і слабкі місця СУБД (наприклад, оптимізатор запитів). Крім робіт, які ведуть фахівці "Microsoft", у цій галузі помітні зусилля дослідників і розробників компанії "Oracle" [8].

Звичайно, наявні та очікувані засоби на виконання таких робіт допомагають адміністраторам баз даних та налаштувачам додатків. Однак їх наявність не обов'язково дозволяє суттєво скоротити кількість "важелів управління", які надають адміністратори з боку СУБД. Автоматичне налаштування баз даних без участі людини-експерта, як і раніше, не підтримує жодна система, а від адміністраторів потрібна ще більш висока кваліфікація. Отже, прогноз щодо можливості забезпечити легку та швидку самоналаштовуваність СУБД не можна вважати здійсненим, хоча він залишається дороговказом для виробничих дослідників.

*Переосмислення традиційної архітектури систем баз даних*. У світлі появи появи середовища, яке буде доступним у найближчому майбутньому, виникла потреба у переосмисленні традиційної архітектури систем баз даних.

Це твердження співзвучне назві статті [1] "Кінець архітектурної епохи, або настав час повністю переписувати системи управління даними". Та й вийшла ця стаття в 2007 р., якраз до кінця першого десятиліття XXI століття. Однак не можна стверджувати, що автори пропонують повністю переосмислені архітектури СУБД. Вони переосмислюють їх у зв'язку з переходом від архітектур універсальних СУБД до архітектур спеціалізованих систем. Компоненти архітектур залишаються відомими та традиційними.

Та загалом і підстав для повного перегляду архітектури у 2000-х роках не було. Тоді відбувались переважно кількісні, а не якісні зміни (суттєво збільшені розміри основної пам'яті, кластери з дуже великою кількістю вузлів тощо.). Незмінною залишалася опора SQL-орієнтованих СУБД на жорсткі диски з рухомими головками. (Цитата зі звіту щодо зустрічі в Асіломарі: "У майбутньому можуть виявитися недоречними архітектури, що "приховують наявність рухливих магнітних головок", такі як RAID 5" – з позицій оптимізації запитів розробникам СУБД завжди були підозрілі пристрої зовнішньої пам'яті, які приховують свою організацію.)

Нижче наведено перелік того, що, на думку авторів, може справді вплинути на архітектуру майбутніх СУБД (нові апаратні можливості представлено в порядку зростання їхнього впливу на архітектуру СУБД).

*Флеш-пам'ять та твердотільні диски*. Твердотільна зовнішня пам'ять (solid-state disk, SSD) має інші характеристики, ніж традиційна дискова пам'ять на дискових пристроях з рухомими головками (hard disk, HD). Основною відмінністю є те, що у пристроїв SSD відсутні магнітні головки, що механічно переміщаються, і вони дійсно є пристроями прямого доступу – час доступу до будь-якого блоку даних в SSD є однаковим. І навіть без урахування часу переміщення головок HD SSD показують час обміну, в кілька разів менший, ніж HD.

Раніше SSD програвали HD за максимально доступним обсягом, вартістю в розрахунку на гігабайт даних та зносостійкістю. Наразі вже доступні SSD ємністю в кілька терабайт. Показники вартості та зносостійкості поки що поступаються HD, але згодом і вони будуть покращені.

Звичайно, виникає бажання перевести наявні СУБД на платформу SSD, але але такі роботи не виконуються. Особливості SSD змушують розробників переглянути алгоритми, які застосовують у багатьох компонентах SQL-орієнтованих СУБД, а можливо й архітектуру СУБД загалом. Наприклад, при переході від SSD до HD перестають ефективно працювати методи кешування блоків бази даних в основній пам'яті [9], що вимагає їх принципового перероблення.

Безумовно, перехід до SSD вимагає значної обробки оптимізатора запитів, починаючи, мабуть, з евристик "відсіювання" свідомо непридатних планів запитів. Більше того, у наборі планів для цього запиту можуть з'явитися плани, які взагалі не генерувалися під час використання HD. Цілком змінюється компонент оцінки вартості плану, оскільки тепер обміни із зовнішньою пам'яттю коштуватимуть набагато дешевше, і ця вартість не залежатиме від розташування блоку даних у зовнішній пам'яті.

Основні вендори SQL-орієнтованих СУБД побоюються цих змін, що зачіпають внутрішні частини СУБД, і тому використовують SSD, в основному нехтуючи їх відмінностями від HD.

*Енергонезалежна основна пам'ять*. Цей вид пам'яті, яка називається non-volatile memory (довготривала, або енергонезалежна, пам'ять), а також storage class memory (пам'ять класу зберігання даних), нарешті стають доступними. Ще 30 років тому використання енергонезалежної пам'яті передбачалося у проєкті системи зберігання Postgres [10]. Майкл Стоунбрейкер хотів використати таку пам'ять (у невеликому обсязі) для зберігання частини кешу блоків бази даних і журналу. Але енергонезалежна пам'ять ще не існувала ні тоді, ні в наступні десятиліття – вона з'явилась тільки в останні роки.

Наразі пам'ять класу зберігання даних стала реальністю. Є кілька варіантів фізичного виконання такої пам'яті, але вони забезпечують пряму адресацію і збереження даних після відключення електроживлення.

Зрозуміло, що використання енергонезалежної пам'яті у СУБД забезпечує значні переваги. Сьогодні досить популярними є СУБД, що зберігають дані у традиційній основній пам'яті (in-memory DBMS). Але такі системи не можуть працювати без дискової пам'яті, і оскільки всі транзакції змінюють дані, повинна забезпечуватися гарантована безпека результатів (durability), тобто ці результати так чи інакше мають бути відображені у зовнішній пам'яті. Через це системи in-memory зазвичай забезпечують істотно більшу продуктивність, ніж традиційні СУБД, для транзакцій, які включають тільки читання. Винятком є масивно-паралельна архітектура H-Store/VoltDB [2, 3], в якій тривалість транзакцій підтримується за рахунок реплік.

*Масивно-мультипотокові архітектури комп'ютерів*. Сьогоднішні комп'ютери розраховані на те, що виконувані в них програми дотримуються принципу локальності. Це означає, що у будь-який момент виконання будь-якого процесу йому достатньо забезпечити деякий обмежений набір команд і даних (робочий набір), причому якщо процес має робочий набор WS в момент часу $T$, то з великою ймовірністю цей робочий набір збережеться і в момент часу $T+t$ для певного значення $t$. Принцип локальності дозволяє підтримувати в процесорах апаратний кеш, і саме наявність цього кешу дає можливість згладити розрив у швидкості між процесором і основною пам'яттю.

Проте є класи додатків, у яких програмам не властивий принцип локальності (додатки біоінформатики, аналізу соціальних мереж та ін.). При роботі таких програм наявність кешу ніяк не сприяє ефективності, і вони працюють фактично зі швидкістю основної пам'яті. Ідея апаратної архітектури для підтримки таких програм з'явилася наприкінці минулого десятиліття в компанії "Cray Inc." [13].

Сутність ідеї полягає в тому, що потрібно зробити масивно-багатопотоковий процесор, який підтримує на апаратному рівні десятки тисяч потоків із загальним доступом до основної пам'яті. Кеш у процесорі не підтримується, але для кожного потоку забезпечується власний набір регістрів. Якщо в певний момент часу виконується потік Tr1 і в ньому був потрібний обмін з основною пам'яттю, то він апаратно блокується і запускається один із потоків Tri, у якого всі необхідні дані знаходяться в регістрах. При досить значній кількості потоків готовий до виконання потік завжди знайдеться, і багатопотокова програма в цілому виконуватиметься зі швидкістю процесора.

Для досягнення таких результатів потрібно навчитися розпаралелювати програму на дуже велику кількість невеликих (кілька десятків команд) потоків. Це дуже складне завдання, відсутність розв'язання якого гальмує розвиток масивно-мультипотокових архітектур. Але потенційна можливість побудови такої архітектури має стимулювати дослідження в галузі баз даних. Припустимо, що масивно-мультипоточний комп'ютер оснащений незалежною основною пам'яттю, і ця пам'ять використовується для зберігання баз даних. СУБД працює на деякому зовнішньому комп'ютері з традиційною архітектурою і при компіляції операторів SQL розпаралелює їх на велику кількість

потоків. Це набагато реальніше, ніж для довільної програми, оскільки SQL є декларативною мовою.

Звичайно, це дуже непроста задача, але якщо її розв'язати, ми зможемо отримати системи баз даних, які опрацьовують запити зі швидкістю кешу.

***6. Зустріч у м. Лоуелл 2003 року.***

*Інтеграція тексту, даних, коду та потоків.* На зустрічі було зазначено, що настав час припинити вбудовувати нові конструкції у стару реляційну архітектуру. Потрібно переосмислити базову архітектуру СУБД із метою підтримки структурованих даних; текстових, просторових, темпоральних та мультимедійних даних; процедурних даних, тобто типів даних та методів їх інкапсуляції; тригерів; потоків і черг даних як рівноправних компонентів першого гатунку всередині архітектури СУБД (як на рівні інтерфейсів, так і на рівні реалізації). У зв'язку з цим фахівці в галузі баз даних мали виробити нову систему понять.

Але «нова система понять» так і не була сформована, а спроб переосмислення базової архітектури універсальних СУБД в минулі роки не було. З одного боку, основні вендори комерційних СУБД продовжували нарощувати функціональні можливості своїх продуктів, намагаючись якнайменше змінювати ядро систем (тобто фактично роблячи нові компоненти об'єктами другого гатунку). З іншого боку, з 2005 р. галузь баз даних перебуває під вливом слогана Майкла Стоунбрейкера "Один розмір непридатний для всіх" [5], на тлі якого було виконано кілька більш-менш вдалих проєктів спеціалізованих СУБД, головною темою яких були саме поняття, на підтримку яких орієнтувалася система.

Як повчальний приклад спроби розширити кількість об'єктів "першого гатунку" в універсальних СУБД варто згадати порівняно недавню історію підтримки XML у базах даних. Специфікація XML (eXtendable Markup Language) з'явилася на початку 1998 р. Ця мова в основному призначалася для подання повідомлень, що передаються в Internet. Очікувалося, що таких повідомлень буде дуже багато, і тому корисно вміти їх зберігати і шукати в колекціях XML-документів. Це було поштовхом до розробления спеціалізованих XML-орієнтованих СУБД із власними системами зберігання та підтримкою мови запитів XQuery. Першими цю роботу розпочали компанія "Software AG" (Tamino [14]) та Інститут системного програмування РАН (Sedna [15]).

Tamino та Sedna були вже порівняно працездатними системами, коли їх розробленням зацікавилися "IBM" та "Oracle". У SQL-орієнтованій СУБД XML з власною мовою запитів підтримувати не так просто, і вендорам фактично довелося реалізувати окреме середовище зберігання XML, процесор і оптимізатор мови XQuery, а потім інтегрувати їх із середовищем SQL, забезпечивши можливість вбудовування XQuery в SQL, SQL – у XQuery і т. д. Потрібно було витратити багато зусиль і коштів, щоб зробити XML і XQuery "повноправними жителями" SQL-орієнтованої СУБД. Обидві компанії виконали цю роботу повністю. Але на час досягнення готовності до використання таких інтегрованих систем інтерес до XML згас. Новим фаворитом стала мова JSON, а результати виконаної роботи (як у галузі створення спеціалізованих XML-орієнтованих систем, так і в напрямі розроблення вбудованих СУБД) виявилися багато в чому незатребуваними.

Але на створення спеціалізованих систем було витрачено менше сил і коштів. Отже, варто подумати про доцільність нової системи понять для універсальних СУБД.

*Об'єднання інформації.* У Internet парадигма ETL неприйнятна. Зараз потрібно проводити інтеграцію інформації між кількома підприємствами. В результаті буде потрібна інтеграція, можливо, мільйонів інформаційних джерел "на льоту".

У такій загальній постановці вирішити проблему навряд чи вдалося. Реалістичний підхід було запропоновано у 2005 р. [16]. У статті стверджувалося, що уніфікований підхід до інтеграції даних неможливий і не потрібний. Повинен забезпечуватися рівень інтеграції, який потрібно конкретному типу додатків. Наприклад, для виконання інтелектуального аналізу даних досить фізично зібрати дані з різних джерел без їх перетворення, а для класичного OLAP потрібна побудова сховища даних з повною підтримкою ETL. На жаль, на практиці гарна ідея просторів даних поки що не була затребувана.

Якщо продовжувати тему ETL і OLAP, то останніми роками порівняно життєздатною

виявилася ідея віртуальних сховищ даних [17], у яких підтримуються великі метадані та інкрементально оновлювані агрегати, а доступ до фактів здійснюється "на льоту" у відповідних зовнішніх джерелах. У таких "сховищах даних" не підтримуються історичні дані, і можливості OLAP обмежені. За це та інші об'єктивні недоліки віртуальні сховища даних досить різко критикує класик технології сховищ даних Білл Інмонн [18].

**7. *Зустріч у отелі "Claremont Resort" 2008 року.*** Зустріч вівдбувалась на тлі підвищеного ажіотажу навколо *"великих даних"*. Наявність проблеми "великих даних" призводить до швидкого зростання кількості користувачів традиційних систем управління базами даних (СУБД), а також стимулює розроблення нових спеціалізованих рішень управління даними на основі спрощених компонентів. Повсюдне використання "великих даних" призводить до зростання числа розробників технологій управління даними, що, безсумнівно, викликає корінну реорганізацію цієї галузі.

З такою думкою учасників зустрічі в отелі "Claremont Resort" не можна не погодитися, хоча вже у 2008 р. вона виглядала не прогнозом, а скоріше спостереженням за тим, що відбувається. До цього часу вже існували спеціалізовані масивно-паралельні аналітичні СУБД від компаній "Vertica" [3], "Greemplum" та "Aster Data" [1], спеціалізована потокова система від компанії "StreamBase" [5], був готовий прототип спеціалізованої масивно-паралельної транзакційної СУБД H-Store [2] її комерціалізації (VoltDB [4]) тощо. Нові спеціалізовані рішення розробляються й у нинішній час [19].

Цікаво, що майже всі вищезгадані компанії були поглинені більш потужними і провідними компаніями, які в основному не спеціалізуються на виробництві продуктів управління базами даних: "Vertica" була поглинена в 2011 р. компанією "Hewlett-Packard": "Greemplum" – в 2010 р. компанією "EMC"; "Aster Data" – у 2011 р. компанією "Teradata"; "StreamBase" – у 2013 р. компанією "TIBCO". Незважаючи на використання передового підходу Майкла Стоунбрейкера утворилась лише низка вдалих стартапів, а реальна технологія управління базами даних, як і раніше, знаходиться в руках традиційних гігантів.

Що стосується другої частини спостереження авторів звіту, то вже в 2008 р. напрям NoSQL розвивався небувалими в історії баз даних темпами. Швидко зростала кількість реалізованих (майже з відкритими вихідними кодами) систем, у яких заперечувалися SQL і ACID-транзакції. У 2006 році розпочався проєкт Hadoop для відкритої реалізації MapReduce. Сьогоднішній список нереляційних, розподілених, горизонтально масштабованих СУБД із відкритими текстами налічує понад 230 систем [21], Hadoop широко використовується, розвивається та вдосконалюється.

Важко сказати, чи означає це корінну реорганізацію галузі баз даних. Сьогодні співтовариство NoSQL уникає як дослідницької роботи, так і участі у традиційній дослідницькій спільноті дослідників баз даних. Вже кілька років проводяться спеціалізовані конференції, присвячені винятково проблематиці NoSQL, але якщо подивитися на сайти цих конференцій, можна побачити, що вони відрізняються від конференцій, що проводяться компаніями "VLDB", "Data Engineering", "SIGMOD" та ін., на яких дослідники розповідають про отримані результати. Вони більше схожі на конференції, які вендори програмних систем проводять для своїх користувачів.

*Зміни в архітектурі комп'ютерних систем*. На макрорівні фундаментальні зміни в архітектурі програмного забезпечення відбуваються завдяки розвитку "хмарних" комп'ютерних сервісів. На мікрорівні в комп'ютерних архітектурах закон Мура зараз трактується на користь не підвищення тактової частоти мікропроцесорів, а збільшення числа процесорних ядер і потоків управління в одному кристалі. Основні зміни в технології зберігання даних стосуються ієрархії пам'яті у зв'язку з доступністю більшого числа кешів збільшеного об'єму на одному кристалі, дешевшої основної пам'яті великого об'єму і флеш-пам'яті.

Із твердженням щодо макрорівня в цілому не можна не погодитись. Модель "Програмне забезпечення як послуга" (Software as a Service, або SaaS) у більшості випадків дозволяє користувачам позбутися потреби у встановленні, адмініструванні та підтримці програмного забезпечення. Звичайно, це стосується і хмарних СУБД. Але з ними пов'язана проблема, на яку не звертають увагу фахівці як із хмарних обчислень, так і з баз даних.

Основою хмарних обчислень є віртуалізація. Користувач запитує у постачальника

хмарних послуг послугу з необхідними характеристиками (на основі Угоди про рівень надання послуги – Service Level Agreement, SLA), і його не стосується те, в який спосіб і за рахунок яких фізичних ресурсів ця слуга надаватиметься. Віртуалізація на всіх рівнях, від рівня IAAS (інфраструктура як послуга) до рівня SaaS передбачає, зокрема, що хмарна СУБД працює на віртуальному сервері, оснащеному віртуальною системою зберігання.

Водночас оптимізатор запитів цієї хмарної СУБД вважає, що СУБД, як і раніше, працює з реальними магнітними дисками з рухомими головками і обирає плани запитів, виконання яких вимагатиме меншої кількості обмінів з дисковими пристроями. Результати можуть виявитися непередбачуваними і дуже неприємними для користувачів. Наразі існує лише один реальний шлях усунення цієї проблеми – за запитом послуги хмарної СУБД забезпечувати їй реальну, а не віртуальну апаратуру з характеристиками користувача. Можливо, це й не знадобиться, але публікації, що це підтверджують, відсутні.

Що стосується мікрорівня, то розпаралелювання запитів у симетричних мультипроцесорних системах (системах із загальною основною пам'яттю), до яких належать і сучасні багатоядерні та частково багатопотокові комп'ютери, підтримується в розвинених SQL-орієнтованих СУБД не перший десяток років. Реальною проблемою є те, що СУБД не може забезпечити горизонтальну масштабованість при зростанні кількості ядер (або ниток) у процесорі через обмеження паралельного доступу до основної пам'яті.

Певну надію на можливість вирішення цієї проблеми дає публікація [20]. У ній пропонується ескіз архітектури СУБД зі зберіганням даних в основній пам'яті, яка начебто забезпечує горизонтальну масштабованість. Загальна ідея полягає в тому, що у системі з n ядрами виконується процес СУБД з n потоками, жорстко прив'язаними до ядер. У кожній нитці виконується вся СУБД цілком, як і у масивно-паралельних СУБД без загальних ресурсів, а поділ даних виконується за допомогою механізму віртуальної пам'яті. Потоки взаємодіють шляхом обміну повідомленнями на основі загальної пам'яті. В разі потреби знову розділити базу даних перепис даних здійснювати не потрібно, змінюється лише таблиця сторінок віртуальної пам'яті. Компіляція та оптимізація запитів виконуються у зовнішньому комп'ютері.

До цього можна додати лише те, що доступність високих обсягів основної пам'яті активізувала розробки СУБД класу in-memory.

**8.** ***Зустріч у Бекманському університетському центрі 2013 року.*** У центрі уваги учасників Бекманської зустрічі була проблема "великих даних".

Фахівці в галузі баз даних мають унікальні можливості для вирішення проблеми "великих даних" і величезний потенціал для революційного впливу на створення технологій збирання та оброблення структурованих і неструктурованих даних із різних джерел. Для реалізації цього потенціалу потрібно звернути особливу увагу на п'ять галузей досліджень:

- масштабовані інфраструктури великих/швидких даних;
- подолання різнорідностей ландшафту управління даними;
- наскрізне оброблення та інтерпретація даних;
- хмарні служби;
-управління різними ролями людей у життєвому циклі даних.

Наразі ще рано коментувати прогнози, що містяться в цьому звіті, і оцінювати, чи справді дослідження проблеми "великих даних" здійснюються саме у зазначених п'яти галузях. Потрібно дочекатися принаймні наступної зустрічі. Але слід зазначити, що створення методів і засобів наскрізного оброблення даних (від необроблених даних до придатних для практичного використання знань) видається дуже об'ємним завданням, що виходить за межі традиційних досліджень.

У зв'язку з цим дуже важливе значення має порушена на цій зустрічі проблема культури досліджень у всьому світі. В останні роки тривожним явищем стало зростання уваги до публікацій та лічильників цитування, а не до результатів досліджень. Це заважає фахівцям виконувати великі дослідницькі проєкти або проводити якісні конференції. Публікації та доповіді на конференціях стають не засобом ознайомлення колег із власними досягненнями та результатами, що приносить задоволення авторам, а лише рутинним тягарем.

На жаль, із цим постійно доводиться стикатися і українським дослідникам у різних

наукових галузях. Вже виросло ціле покоління дослідників, які ніколи не писали статті й не виступали на конференціях лише тому, що вони не відчули в цьому потребу. Учасники Бекманської зустрічі не дійшли згоди щодо способів вирішення цієї проблеми, але очевидно, що потрібно прагнути до того, щоб написання та публікація статті приносили авторам радість, а не сприймались ними як нав'язані ззовні додаткові неприємні обов'язки.

**Висновки**. Звіти про зустрічі фахівців у галузі баз даних дуже корисні для всіх тих, хто цікавиться технологією баз даних. Вони дають зрозуміти, коли і чим керувалися дослідники у своїй роботі, під впливом яких зовнішніх чинників чи напрямів досліджень вони перебували і які потреби в розвитку загальної технології враховували.

У статті зроблено спробу здійснити ретроспективний погляд у багаторічну історію баз даних після першої зустрічі дослідників у 1988 р. у Лагуна-Біч. Роздуми авторів суб'єктивні, і з ними не обов'язково потрібно погоджуватися.

За результатами проведеного аналізу можна спрогнозувати напрями подальших досліджень у галузі баз і сховищ даних:

1. Підтримка мультимедійних об'єктів
1.1. Третинна пам'ять
1.2. Нові типи даних
1.3. Якість обслуговування
1.4. Запити з нечіткими критеріями
1.5. Підтримка інтерфейсів користувача
2. Розподіл інформації
2.1. Ступінь автономності
2.2. Облік та розрахунки
2.3. Безпека та конфіденційність
2.4. Реплікація та узгодження даних
2.5. Інтеграція та перетворення даних
2.6. Вибірка та виявлення даних
2.7. Якість даних
3. Нові застосування баз даних
3.1. Інтелектуальний аналіз даних
3.2. Сховища даних
3.3. Репозитарії
4. Управління потоками робіт та транзакціями
4.1. Управління потоками робіт
4.2. Альтернативні моделі транзакцій
5. Простота використання

Більшість із них є прикладними, тому їх реалізація впливатиме на технології створення програмних та інтелектуальних систем і подальший розвиток засобів і середовищ програмування.

**Список використаних джерел**

1. Кузнецов С. Д. MapReduce: внутри, снаружи или сбоку от параллельных СУБД? // Труды ИСП РАН. 2010. Т. 19. С. 35–40.
2. Stonebraker M., Madden S., Abadi D. J., Harizopoulos S., Hachem N., Helland P. The End of an Architectural Era (It's Time for a Complete Rewrite) // Proceedings of VLDB. 2007. P. 1150–1160.
3. Vertica Accelerator. The Fastest Analytics and Machine Learning – from Start to Finish. URL: https://www.vertica.com/
4. VoltDB. URL: https://www.voltdb.com/
5. Stonebraker M., Çetintemel U. "One Size Fits All": An Idea Whose Time Has Come and Gone // ICDE '05, Proceedings of the 21st International Conference on Data Engineering, April 5–8, 2005. P. 2–11.
6. Chaudhuri S., Narasayya V. AutoAdmin "what-if" index analysis utility // SIGMOD '98, Proceedings of the 1998 ACM SIGMOD international conference on Management of data, Seattle, Washington, USA, June 1–4, 1998. P. 367–378.
7. Bruno N., Chaudhuri S., König A. C., Narasayya V., Ramamurthy R., and Syamala

M. AutoAdmin Project at Microsoft Research: Lessons Learned // Bulletin of the Technical Committee on Data Engineering. 2011. Vol. 34. No 4. P. 12–19.
8. Belknap P., Beresniewicz J., Dageville B., Dias K., Shaft U., Yagoub K. A Decade of Oracle Database Manageability // Bulletin of the Technical Committee on Data Engineering. 2011. Vol. 34. No 4. P. 20–27.
9. Кузнецов С. Д., Прохоров А. А. Алгоритмы управления буферным пулом СУБД при работе с флэш-накопителями // Труды ИСП РАН. 2012. Т. 23. С. 173–194. https://doi.org/10.15514/ISPRAS-2012-23-11
10. Stonebraker M. The Design of the POSTGRES Storage System // VLDB '87, Proceedings of the 13th International Conference on Very Large Data Bases, September 01–04, 1987. P. 289–300.
11. DeBrabant J., Arulraj J., Pavlo A., Stonebraker M., Zdonik S. B., Dulloor S. A Prolegomenon on OLTP Database Systems for Non-Volatile Memory // ADMS 2014, Fifth International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures, September 1, 2014. P. 57–63.
12. Oukid I., Lehner W. Towards a Single-Level Database Architecture on Non-Volatile Memory (Presentation Abstract) // 8th Annual Non-Volatile Memories Workshop 2017, University of California, San Diego – Price Center Ballroom East, March 12–14, 2017.
13. Tumeo A., Secchi S., and Villa O. Designing Next-Generation Massively Multithreaded Architectures for Irregular Applications // Computer. 2012. Vol. 45. No 8. P. 53–61.
14. Schöning H. Tamino – A DBMS Designed for XML // ICDE '01, Proceedings of the 17th International Conference on Data Engineering, April 02–06, 2001. P. 149.
15. Fomichev A., Grinev M., Kuznetsov S. Sedna: A Native XML DBMS // Lecture Notes in Computer Science. 2006. Vol. 3831. P. 272–281. https://doi.org/10.1007/11611257_25
16. Franklin M., Halevy A., Maier D. From Databases to Dataspaces: A New Abstraction for Information Management // SIGMOD Record. 2005.Vol. 34. No 4. P. 27–33.
17. Virparia P. V., Buch S. H., Parabia R. F. Trade and Tricks: Traditional vs. Virtual Data Warehouse // An International Journal of Advanced Engineering & Applications. 2010. Vol. 2. No 3. P. 225–239.
18. Inmon B. The Elusive Virtual Data Warehouse, March 19, 2009. URL: https://www.academia.edu/562634/Trade_and_Tricks_Traditional_vs_Virtual_Data_Warehouse
19. Gadepally V., Chen P., Duggan J., Elmore A., Haynes B., Kepner J., Madden S., Mattson T. Stonebraker M. The BigDAWG Polystore System and Architecture // Proceedings of 2016 IEEE High Performance Extreme Computing Conference (HPEC). P. 1–6.
20. Pandis I., Johnson R., Hardavellas N., Ailamaki A. Data-oriented transaction execution // Proceedings of the VLDB Endowment. 2010. Vol. 3. No 1. P. 928–939.
21. Кузнецов С. Д. Управление данными: 25 лет прогнозов // Труды Института системного программирования РАН. 2017. Т. 29. Вып. 2. С. 117–160.

**References**

1. Kuznetsov S. D. (2010). MapReduce: vnutri, snaruzhi ili sboku ot parallelnyih SUBD? [MapReduce: within, outside, or on the side-by-side with parallel DBMSs?]. *Trudy ISP RAN – Proceedings of the Institute for System Programing of the Russian Academy of Sciences*, 19, 35–40 [in Russian].
2. Stonebraker M., Madden S., Abadi D. J., Harizopoulos S., Hachem N., & Helland P. (2007). The End of an Architectural Era (It's Time for a Complete Rewrite). *Proceedings of VLDB*, 1150–1160.
3. Vertica Accelerator. The Fastest Analytics and Machine Learning – from Start to Finish. Retrieved from https://www.vertica.com/
4. VoltDB. Retrieved from https://www.voltdb.com/
5. Stonebraker M. & Çetintemel U. (2005). "One Size Fits All": An Idea Whose Time Has Come and Gone. *ICDE '05, Proceedings of the 21st International Conference on Data Engineering*, 2–11.

6. Chaudhuri S., & Narasayya V. (1998). AutoAdmin "what-if" index analysis utility. *SIGMOD '98, Proceedings of the 1998 ACM SIGMOD International Conference on Management of Data*, 367–378.
7. Bruno N., Chaudhuri S., König A. C., Narasayya V., Ramamurthy R., & Syamala M. (2011). AutoAdmin Project at Microsoft Research: Lessons Learned. *Bulletin of the Technical Committee on Data Engineering*, vol. 34, issue 4, 12–19.
8. Belknap P., Beresniewicz J., Dageville B., Dias K., Shaft U., & Yagoub K. (2011). A Decade of Oracle Database Manageability. *Bulletin of the Technical Committee on Data Engineering*, vol. 34, issue 4, 20–27.
9. Kuznetsov S. D., & Prokhorov A. A. (2012). Algoritmyi upravleniya bufernyim pulom SUBD pri rabote s flesh-nakopitelyami [Algorithms for control of DBMS buffer pool in handling flash drives]. *Trudy ISP RAN – Proceedings of the Institute for System Programing of the RAS*, 23, 173–194. https://doi.org/10.15514/ISPRAS-2012-23-11 [in Russian].
10. Stonebraker M. (1987). The Design of the POSTGRES Storage System. *VLDB '87, Proceedings of the 13th International Conference on Very Large Data Bases*, 289–300.
11. DeBrabant J., Arulraj J., Pavlo A., Stonebraker M., Zdonik S. B., & Dulloor S. (2014). A Prolegomenon on OLTP Database Systems for Non-Volatile Memory. *ADMS 2014, Fifth International Workshop on Accelerating Data Management Systems Using Modern Processor and Storage Architectures*, 57–63.
12. Oukid I., & Lehner W. (2017). Towards a Single-Level Database Architecture on Non-Volatile Memory (Presentation Abstract). *8th Annual Non-Volatile Memories Workshop 2017, University of California*.
13. Tumeo A., Secchi S., & Villa O. (2012). Designing Next-Generation Massively Multithreaded Architectures for Irregular Applications. *Computer*, vol. 45, issue 8, 53–61.
14. Schöning H. (2001). Tamino – A DBMS Designed for XML. *ICDE '01, Proceedings of the 17th International Conference on Data Engineering*, p. 149.
15. Fomichev A., Grinev M., & Kuznetsov S. (2006). Sedna: A Native XML DBMS. *Lecture Notes in Computer Science*, 3831, 272–281. https://doi.org/10.1007/11611257_25
16. Franklin M., Halevy A., & Maier D. (2005). From Databases to Dataspaces: A New Abstraction for Information Management. *SIGMOD Record*, vol. 34, issue 4, 27–33.
17. Virparia P. V., Buch S. H., & Parabia R. F. (2010). Trade and Tricks: Traditional vs. Virtual Data Warehouse. *An International Journal of Advanced Engineering & Applications*, 2010, vol. 2, issue 3, 225–239.
18. Inmon B. (2009). *The Elusive Virtual Data Warehouse*, March 19. Retrieved from https://www.academia.edu/562634/Trade_and_Tricks_Traditional_vs_Virtual_Data_Warehouse
19. Gadepally V., Chen P., Duggan J., Elmore A., Haynes B., Kepner J., et al. (2016). The BigDAWG Polystore System and Architecture. *Proceedings of 2016 IEEE High Performance Extreme Computing Conference (HPEC)*, 1–6.
20. Pandis I., Johnson R., Hardavellas N., & Ailamaki A. (2010). Data-oriented transaction execution. *Proceedings of the VLDB Endowment*, vol. 3, issue 1, 928–939.
21. Kuznetsov S. D. (2017). Upravlenie dannyimi: 25 let prognozov [Data management: 25 years of forecasts]. *Trudy ISP RAN – Proceedings of the Institute for System Programing of the Russian Academy of Sciences*, 2017, 29(2), 117–160 [in Russian].

**Посилання на статтю:**

Єршова О. Л., Ставицький О. В. Дослідження сучасних тенденцій у технологіях баз і сховищ даних як технологічної та архітектурної основи для створення програмних і інтелектуальних систем засобами сучасних мов програмування. Частина 2. Науковий вісник Національної академії статистики, обліку та аудиту: зб. наук.пр.. 2022. №1-2. С.76-85. doi: 10.31767/nasoa.1-2-2022.09.