Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)
Кафедра комп'ютерних наук
(повна назва кафедри)

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня

бакалавр
(назва освітнього ступеня)

на тему: Аналіз методів підвищення продуктивності засобів опрацювання великих даних

Виконав: студент IV курсу, групи СНс-42
спеціальності 122 Комп'ютерні науки
(шифр і назва спеціальності)

_____ Лебідко Д.М.
(підпис) (прізвище та ініціали)

Керівник _____ Пасічник В.В.
(підпис) (прізвище та ініціали)

Нормоконтроль _____ Шимчук Г.В.
(підпис) (прізвище та ініціали)

Завідувач кафедри _____ Боднарчук І.О.
(підпис) (прізвище та ініціали)

Рецензент _____ Стадник М.А.
(підпис) (прізвище та ініціали)

Тернопіль
2022

Міністерство освіти і науки України
Тернопільський національний технічний університет імені Івана Пулюя

Факультет комп'ютерно-інформаційних систем і програмної інженерії
(повна назва факультету)

Кафедра комп'ютерних наук
(повна назва кафедри)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ Боднарчук І.О.
(підпис) (прізвище та ініціали)
«_22_» _червня_ 2022 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня Бакалавр
(назва освітнього ступеня)

за спеціальністю 122 Комп'ютерні науки
(шифр і назва спеціальності)

Студенту Лебідко Дмитру Миколайовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Аналіз методів підвищення продуктивності засобів опрацювання великих даних

Керівник роботи Пасічник В.В., д.т.н., професор кафедри КН
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Затверджені наказом ректора від « 16 » березня 2022 року № 4/7-161

2. Термін подання студентом завершеної роботи 21 червня 2022р.

3. Вихідні дані до роботи Наукові публікації про обробку великих даних та методи підвищення продуктивності їх аналітичного опрацювання

4. Зміст роботи (перелік питань, які потрібно розробити)
Вступ. 1 Аналіз предметної області BigData. 1.1 Великі за обсягом дані. 1.2 Великі дані та аналітика великих даних. 1.3 Архітектури для аналітики великих даних. 1.4 Бенчмаркінг великих даних. 1.5 Огляд налаштування систем БД. 2. Аналіз методів підвищення продуктивності засобів опрацювання великих даних. 2.1 Підходи до підвищення продуктивності засобів опрацювання великих даних. 2.2 Налаштування великомасштабних систем обробки BigData. 2.3 Розміщення сховища та розміщення даних. 2.4 Тиражування даних, передача даних, відмовостійкість та продуктивність BigData. 2.5 Кешування та налаштування пам'яті. 2.6 Сховища великих даних і логічна організація. 2.7 Рекомендації для підвищення продуктивності засобів аналітичного опрацювання великих даних. 3. Безпека життєдіяльності, основи охорони праці. Висновки. Перелік джерел.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, слайдів)
1 Титульна сторінка. 2 Тема та мета роботи. 3 Завдання роботи. 4 Актуальність роботи. 5. Екосистема Hadoop. 6 MapReduce.7 Система обміну даними Apache Kafka. 8 Компоненти архітектури великих даних загального призначення. 9 Архітектура Лямбда. 10 Архітектура «Каппа». 11 Розподілені запити до гетерогенних баз даних. 12 Контрольні показники операцій. 13 Методи підвищення продуктивності. 14 Рекомендації для підвищення продуктивності. 15 Висновки. 16 Завершальний слайд.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Безпека життєдіяльності, основи охорони праці	Гурик Олег Ярославович, доцент	04.04.2022	01.05.2022

7. Дата видачі завдання 24 січня 2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1.	Ознайомлення з завданням до кваліфікаційної роботи	24.01.2022	*Виконано*
2.	Підбір джерел про обробку великих даних та методи	04.01.2022-30.01.2022	*Виконано*
	підвищення продуктивності аналітичного		
	опрацювання		
3.	Переклад та опрацювання джерел про обробку	31.01.2022-06.02.2022	*Виконано*
	великих даних та методи підвищення продуктивності		
	аналітичного опрацювання		
4.	Виконання дослідження методів підвищення	07.02.2022-13.02.2022	*Виконано*
	продуктивності аналітичного опрацювання великих		
	даних		
5.	Оформлення розділу «Аналіз предметної області	14.02.2022-06.03.2022	*Виконано*
	BigData»		
6.	Оформлення розділу «Аналіз методів підвищення	07.03.2022-03.04.2022	*Виконано*
	продуктивності засобів опрацювання великих даних»		
7.	Виконання завдання до підрозділу «Безпека життєдіяльності»	04.04.2022-17.04.2022	*Виконано*
8.	Виконання завдання до підрозділу «Основи охорони	18.04.2022-01.05.2022	*Виконано*
	праці»		
9.	Оформлення кваліфікаційної роботи	02.05.2022-15.05.2022	*Виконано*
10.	Нормоконтроль	16.05.2022-22.05.2022	*Виконано*
11.	Перевірка на плагіат	09.06.2022	*Виконано*
12.	Попередній захист кваліфікаційної роботи	10.06.2022	*Виконано*
13.	Захист кваліфікаційної роботи	23.06.2022	

Студент _____ Лебідко Д. М.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Пасічник В.В.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

Аналіз методів підвищення продуктивності засобів опрацювання великих даних // Кваліфікаційна робота освітнього рівня «Бакалавр» // Лебідко Дмитро Миколайович // Тернопільський національний технічний університет імені Івана Пулюя, факультет комп'ютерно-інформаційних систем і програмної інженерії, кафедра комп'ютерних наук, група СНс-42 // Тернопіль, 2022 // С. 50, рис. – 15, табл. – 1, кресл. – 16, бібліогр. – 60.

Ключові слова: аналітичне опрацювання, налаштування продуктивності, платформи великих даних, продуктивність, системи баз даних, системи великих даних.

Кваліфікаційна робота присвячена аналізу методів підвищення продуктивності засобів опрацювання великих даних. Мета роботи підвищення якості надання послуг в галузі аналітичного опрацювання даних.

В першому розділі кваліфікаційної роботи подано означення великих за обсягом даних. Описано великі дані та аналітику великих даних. Розглянуто типові архітектури для аналітики великих даних. Прокласифіковано типи BigDate-систем. Описано типи BigDate-навантаження. Розглянуто BigDate-архітектури. Висвітлено поняття бенчмаркінґу великих даних. Подано огляд налаштування систем баз даних. В другому розділі кваліфікаційної роботи проаналізовано підходи до підвищення продуктивності засобів опрацювання великих даних. Зокрема, розглянуто логічне та фізичне оформлення великих даних. Описано перемикачі та параметри систем. Висвітлено техніки машинного навчання для підвищення продуктивності BigData. Проаналізовано налаштування великомасштабних систем обробки BigData. Розглянуто розміщення сховища та розміщення даних. На основі аналізу обширного переліку наукових публікацій запропоновано рекомендації для підвищення продуктивності засобів опрацювання великих даних по категоріях.

ANNOTATION

Analysis of methods to the productivity increase of big data processing tools // Qualification work of the educational level "Bachelor" // Lebidko Dmytro Mykolaiovych // Ternopil Ivan Pulyuy National Technical University, Computer information systems and software engineering faculty, Computer science department, Group SNs-42 // Ternopil, 2022 // P. 50, fig. - 15, tabl. - 1, chair. - 16, ref. -60.

Keywords: analytical processing, performance tuning, big data platforms, performance, database systems, big data systems.

Qualification work is devoted to the analysis of methods of increase of productivity of means of processing of big data. The purpose of improving the quality of services in the field of analytical data processing.

In the first section of the qualification work the definitions of large data are given. Big data and big data analytics are described. Typical architectures for big data analytics are considered. Types of BigDate systems are classified. Describes the types of BigDate load. BigDate-architecture is considered. The concept of big data benchmarking is covered. An overview of database system settings is provided. In the second section of the qualification work the approaches to increase the productivity of big data processing tools are analyzed. In particular, the logical and physical design of big data is considered. Switches and system parameters are described. Machine learning techniques to increase BigData productivity are highlighted. The settings of large-scale BigData processing systems are analyzed. Storage location and data placement are considered. Based on the analysis of an extensive list of scientific publications, recommendations for improving the productivity of large data processing tools by category are proposed.

ПЕРЕЛІК СКОРОЧЕНЬ І ТЕРМІНІВ

DBA (англ. Database Analyst) – Аналітик баз даних.

HDFS (англ. Hadoop Distributed File System) – Розподілена файлова система Hadoop.

ETL (англ. Extract, Transform, Load) – Витяг, Перетворення та Завантаження.

NP (англ. Neural Process) – Нейронний процес.

OLAP (англ. Online Analytical Processing) – аналітична обробка у режимі реального часу.

OLTP (англ. Online Transaction Processing) – онлайн-обробка транзакцій.

RDBM (англ. Relational database management system) – система управління реляційною базою даних.

SLA (англ. Service Level Agreement) – «Угода про рівень обслуговування».

БД – бази даних.

СУБД – система управління базами даних.

ЗМІСТ

ВСТУП

Актуальність теми. На даний час опубліковано результати багатьох дослідницьких робіт, що стосуються платформ великих даних, які очікують подальших етапів розвитку аналітики даних. Це складні й зазвичай розподілені середовища, що складаються з кількох систем та інструментів. Як і очікувалося, є потреба більш детально розглянути питання продуктивності засобів аналітичного опрацювання великих даних.

Налаштування продуктивності великих даних та NoSQL-сховищ значно відрізняються від так званих звичайних систем. Переважна більшість існуючих рішень – це переважно спеціальні індивідуальні дії, які не підходять для узагальнених ситуацій. Проте на даний час присутні категорії керованих даними рішень, які можна сприймати як установчі принципи та включати в модулі автоматичного налаштування загального призначення для систем великих даних. Тому аналіз методів підвищення продуктивності засобів опрацювання великих даних є актуальним напрямком сучасних досліджень.

Мета і задачі дослідження. Метою даної кваліфікаційної роботи освітнього рівня «Бакалавр» є підвищення якості надання послуг в галузі аналітичного опрацювання даних. Для досягнення поставленої мети потрібно:

– Проаналізувати стан досліджень в галузі аналітичного опрацювання великих даних.

– Дослідити інформаційно-технологічні архітектури для аналітики великих даних.

– Проаналізувати підходи до підвищення продуктивності засобів аналітичного опрацювання великих даних.

Практичне значення одержаних результатів. На основі аналізу обширного переліку наукових публікацій подано огляд налаштування систем БД. В результаті дослідження інформаційно-технологічних архітектур для аналітики великих даних описано типи BigData-систем, BigData-навантаження та поширених на даний час BigData-архітектур. Сформовано перелік рекомендацій для підвищення продуктивності засобів аналітичного опрацювання великих даних.

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ BIGDATA

1.1 Великі за обсягом дані

Для ґенерації великих обсягів розподілених і неоднорідних даних [1] використовується широкий спектр різнотипових пристроїв, зокрема:

– мобільні телефони;

– GPS-пристрої;

– соціальні мережі;

– давачі;

– IoT-пристрої.

Перетворення такої величезної кількості даних у цінну інформацію з одночасним виявленням її основного значення є важливою функцією аналітики великих даних (див. рисунок 1.1) [2].

Рисунок 1.1 – Аналітика великих даних

Нові вимоги з точки зору аналітики, зокрема, управління даними в реальному часі, сприяють створенню середовища аналітики великих даних, що складається з множини рішень і розподілених систем, загально-відомих як стек великих даних [2]. Багато з цих програмно-алгоритмічних рішень передбачають

розподілену обробку великих обсягів даних, які тиражуються або переміщуються по мережах на льоту [3]. Збої трапляються досить часто, і потрібне автоматичне відновлення під час виконання, яке зазвичай досягається на основі повторної обробки завдань і використання проміжного опрацювання даних [4]. У такому середовищі витрати на ввід-вивід і переміщення даних можуть призвести до зростання відповідних та накладних витрат. Планування потужності може виявитися проблемою і призвести до збільшення витрат, особливо в застосунках, які повинні відповідати певному типу «Угоди про рівень обслуговування» (SLA, див. рисунок 1.2), визначеному з точки зору показників продуктивності [2].

Рисунок 1.2 – «Угоди про рівень обслуговування» (SLA)

На даний час немає оптимізатора для створення найефективнішого плану виконання для виконання завдання кожного користувача, що підвищує важливість ретельного налаштування продуктивності систем класу великих даних.

1.2 Великі дані та аналітика великих даних

Термін «Великі дані» широко використовується протягом останніх десяти років у зв’язку зі зростанням використання:

– «розумних» пристроїв;

– соціальних мереж;

– GPS-пристроїв

– а також останнім часом із появою Інтернету речей (IoT) [1].

Ці пристрої та платформи ґенерують величезні обсяги даних у різних форматах, наприклад, текст, аудіо та відео тощо. Великі дані зазвичай асоціюються з так званими ключовими характеристиками [5], зокрема:

– обсяг – стосується розміру даних, що ґенеруються пристроями та платформами, які необхідно зберігати та обробляти;

– різноманітність – стосується різноманітності джерел і моделей даних, наприклад, структурованих, напівструктурованих та неструктурованих даних;

– швидкість – позначає, як швидко створюються дані та як швидко вони повинні бути оброблені, щоб генерувати відповіді в режимі майже реального часу [6].

– достовірність – правдивість даних [7];

– розуміння, яке можна видобути із даних;

– цінність – додана вартості для компаній [8];

– мінливість – значення даних та їх варіантів у різних контекстах, наприклад, просторові та часові [9].

Аналітика даних зосереджена на дослідженні наборів даних, щоб отримати уявлення та зробити висновки щодо інформації, яку вони містять, за допомогою автоматизованих процесів і спеціалізованих систем. Це широка тема, яка охоплює різні типи аналізу даних, наприклад:

– з використанням статистичних методів – описова аналітика;

– пояснення, чому щось сталося – діагностична аналітика;

– що, ймовірно, станеться в майбутньому – прогностична аналітика [6].

Традиційні підходи, зокрема, сховищ даних і OLAP [10], засновані на монолітних і централізованих системах з чітко визначеною організацією. Дані зазвичай створюються та споживаються одним і тим же об’єктом, і вони попередньо обробляються перед завантаженням у сховище даних. Дані та результати запиту вважаються правильними та повними. Водночас, в аналітиці великих даних існує потреба обробки в режимі майже реального часу. Дані часто є напівструктурованими або неструктурованими. Вихідні дані, розподілені на декількох платформах, можуть використовуватися без ґарантії довіри для отримання повних результатів. Зберігання, упорядкування та отримання інформації з дуже великих, гетерогенних і дуже динамічних наборів даних породжують ряд проблем з їх обробкою та управлінням [9].

Зростання використання розподілених систем, потенційно географічно віддалених одна від одної, відновило інтерес до обробки розподілених транзакцій і запитів, а також до компромісів між високою пропускною здатністю, низькою затримкою, пропускною здатністю, узгодженістю, рівнем ізоляції та масштабованістю [1]. Багато систем, керованих даними, перейшли в загальнодоступні хмари, які надають гнучкі та еластичні послуги на вимогу [11]. Тоді обчислювальні ресурси та дані знаходяться в хмарі, але обчислювальні ресурси керуються незалежно від ресурсів зберігання. Це поділ є новинкою, яка впливає на дизайн та оптимізацію хмарних систем управління та аналізу даних.

Розвиток апаратного забезпечення також впливає на архітектуру систем управління даними. Наприклад, бази даних із прискореним графічним процесором мають потенціал для розкладання інтенсивних обчислень, зокрема сортування та агрегації, на підзапити, які можуть виконуватися паралельно [12].

Інтелектуальний аналіз даних і машинне навчання також тісно пов’язані з пошуком та обробкою великих наборів даних [13], тому також було б цікаво дослідити, як використовувати спеціалізовані методи оптимізації запитів для типових робочих навантажень у цих областях.

1.3 Архітектури для аналітики великих даних

1.3.1 Типи BigData-систем

За останні 15 років було запропоновано декілька рішень спеціального призначення для великих даних. Одним із перших був Hadoop [14] – фреймворк для розподіленої обробки великих наборів даних між кластерами.

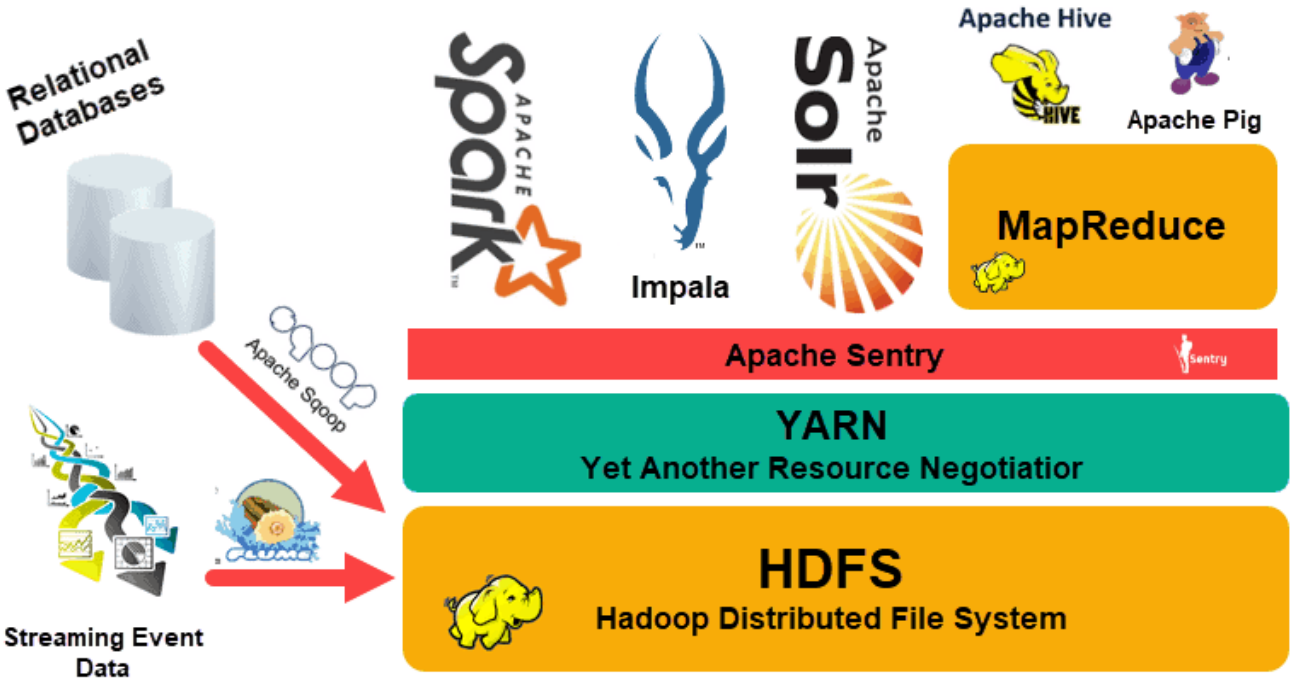

Рисунок 1.3 – Екосистема Hadoop

Hadoop має три основні компоненти:

– файлову систему (HDFS);

– екземпляр MapReduce;

– «Ще один узгоджувач ресурсів» (YARN).

HDFS – це відмовостійка розподілена файлова система, розроблена для розгортання на стандартному обладнанні. За замовчуванням вона використовує великі блоки розміром 128 Мб, які реплікуються і розподіляються між вузлами-учасниками.

«MapReduce» – це парадигма програмування (див. рисунок 1.4) для написання та виконання паралельних програм з функціями «Map» і «Reduce» у кластері.

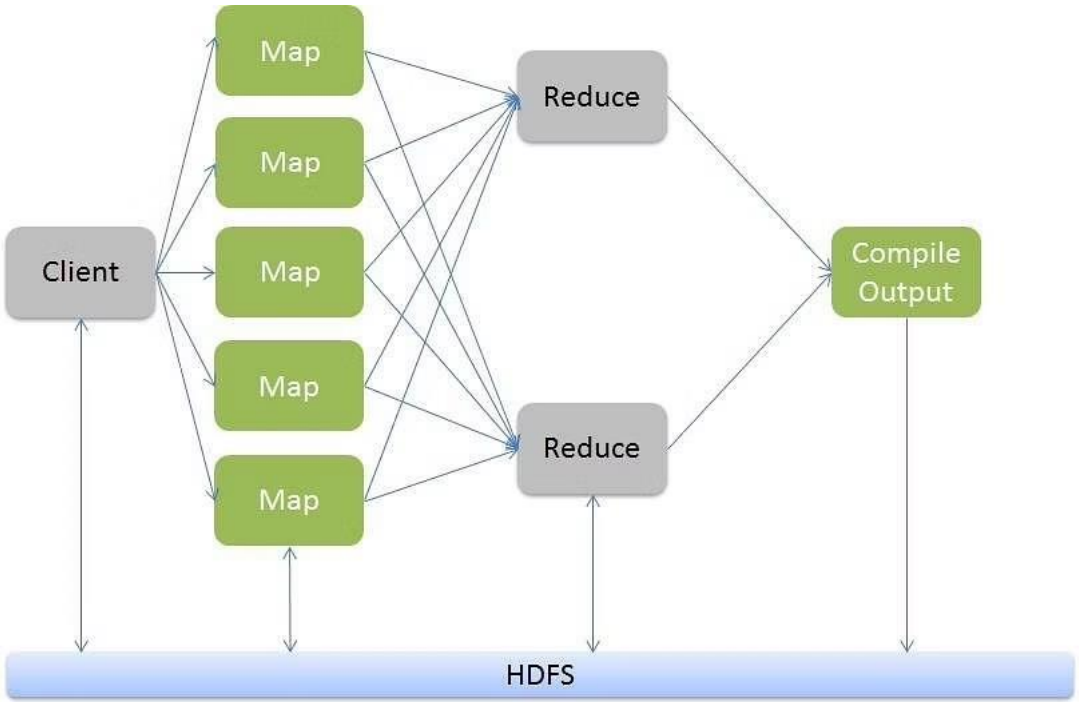

Рисунок 1.4 – Парадигма «MapReduce»

YARN є менеджером ресурсів і відповідає за керування поданими завданнями та розподіл ресурсів для запущених програм.

Наразі існує кілька пропозицій SQL-on-Hadoop, створених на основі Hadoop, зокрема:

– Hive [15];

– Impala [16];

– Spark SQL [17].

Вони були розроблені для запитів OLAP до великих наборів даних [18] і використовують стовпчасті формати зберігання, наприклад, Parquet і ORC. Їм бракує підтримки транзакцій, але останні рішення, зокрема Kudu [19], пропонують базову підтримку для оновлення та видалення даних. Водночас існує декілька категорій NoSQL баз даних, які надають багато різних варіантів для зберігання та пошуку структурованих, напівструктурованих або

неструктурованих великих наборів даних. Серед них сховища ключ-значення [20]:

– «Hazelcast» (див. рисунок 1.5).
– «Redis».
– «Membase/Couchbase».
– «Riak».
– «Voldemort».
– «Infinispan».

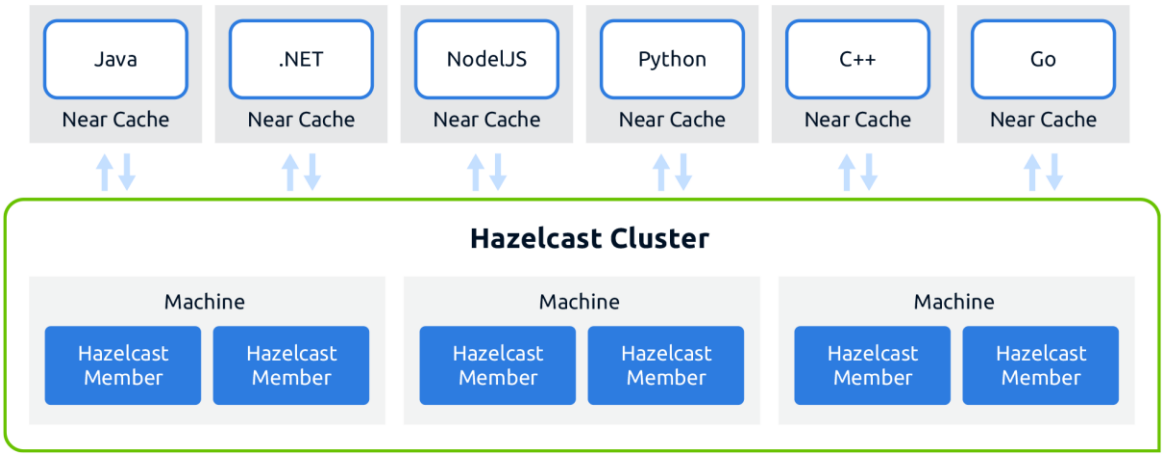

Рисунок 1.5 – Кластер «Hazelcast»

Сховища документів [21]:

– «CouchDB».
– «MongoDB» (див. рисунок 1.6).
– Terrastore.
– RavenDB.
– SimpleDB

Сховища графіків [22]:

– «Neo4J».
– «InfoGrid».
– «HypergraphDB»
– «AllegroGraph»
– «BigData».

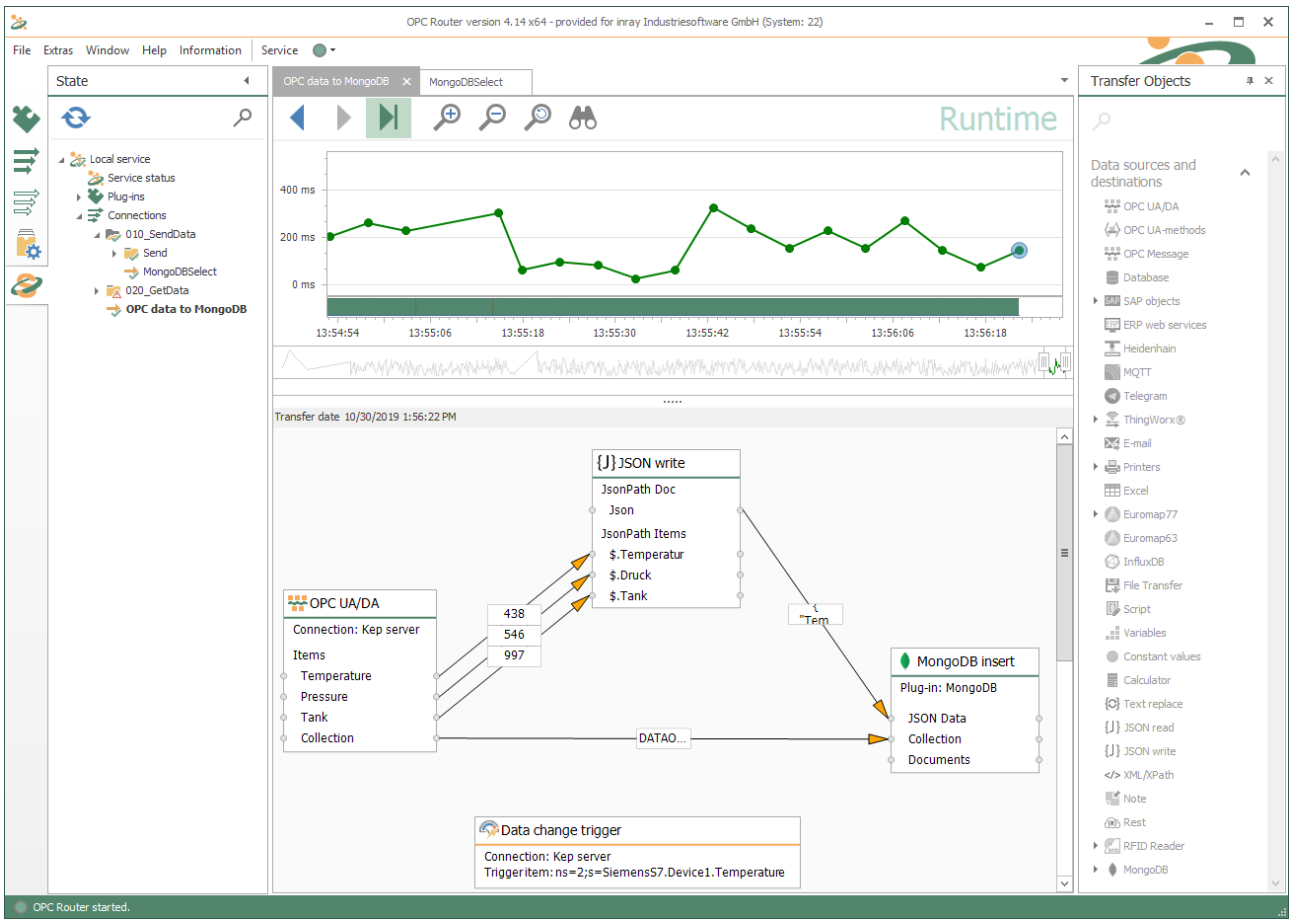

Рисунок 1.6 – Використання «MongoDB» в промисловості та Інтернеті речей

А також потрійні магазини [23]:

– «Virtuoso».

– «Apache Jena».

– «Sesame».

Більшість зазначених платформ зберігають дані на дисках, а швидкість доступу залежить від характеристик дисків. Однак пам'ять набагато дешевша, ніж раніше, і потреба в швидкому доступі до даних спричинила появу сховищ в пам’яті [12]. Бази даних у пам’яті можуть бути набагато швидшими, ніж на диску, і їх легше реалізувати [24]. Існують спеціальні бази даних в пам’яті [25]:

– «Apache Ignite» (див. рисунок 1.7).

– «Gemfire».

– «FluteDB».

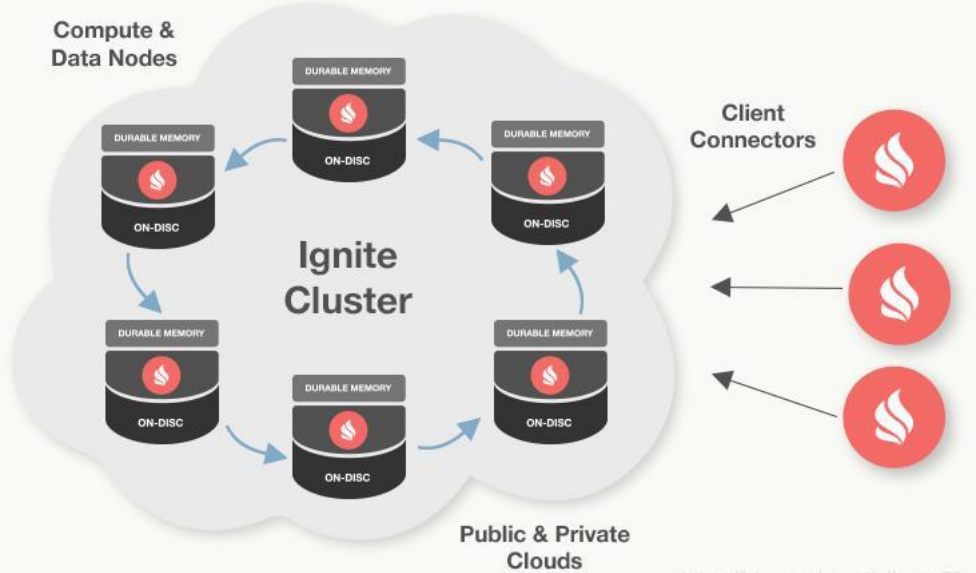

Рисунок 1.7 – Кластер «Apache Ignite»

Але комерційні бази даних, зокрема «Oracle», «Microsoft» і «SAP», також пропонують рішення для керування даними в пам’яті. Основна пам’ять нестабільна, і дані можуть бути втрачені, але впровадження енергонезалежної пам’яті з довільним доступом (NVM) може допомогти подолати цю проблему [26]. Вони майже так само швидкі, як і динамічна пам’ять із випадковим доступом (DRAM). Проте в обох випадках відбуваються важливі архітектурні зміни, які потенційно зводять нанівець деякі традиційні варіанти проектування в системах керування даними [27]. Проблема полягає в тому, що архітектури та обчислювальні моделі у великих даних і високопродуктивних обчисленнях є досить різними, і потрібні нові структури даних, алгоритми та стратегії оптимізації, щоб підвищити продуктивність і масштабованість цих програм.

1.3.2 Типи BigData-навантаження

У системах аналітики великих даних є два основних типи навантаження – пакетна та потокова обробка.

Пакетна обробка полягає в обробці великої кількості даних одночасно. Дані знаходяться в автономному режимі і мають різні формати [12]. Задачі складаються з виконання кількох завдань:

– вилучення даних;

– перетворення;

– завантаження.

Вони виконуються в безперервному та послідовному порядку. Отримані дані також зберігаються в автономному режимі. Цей підхід ефективний для періодичної обробки великих обсягів даних, які збираються протягом заздалегідь визначених періодів часу.

Потокова обробка полягає в обробці та аналізі даних, що надходять з кількох джерел з високою швидкістю, щоб отримувати результати в режимі майже реального часі. Це вимагає безперервного обчислення статичних запитів у ковзному вікні живих вхідних даних [12]. Під час цього процесу вихідні та вхідні потоки можна:

– сортувати;

– фільтрувати;

– об’єднувати;

– нормалізувати;

– збагачувати наявними метаданими.

На даний момент існує декілька платформ для потокової передачі даних, зокрема «Apache Kafka» і «Apache Flink» [28].

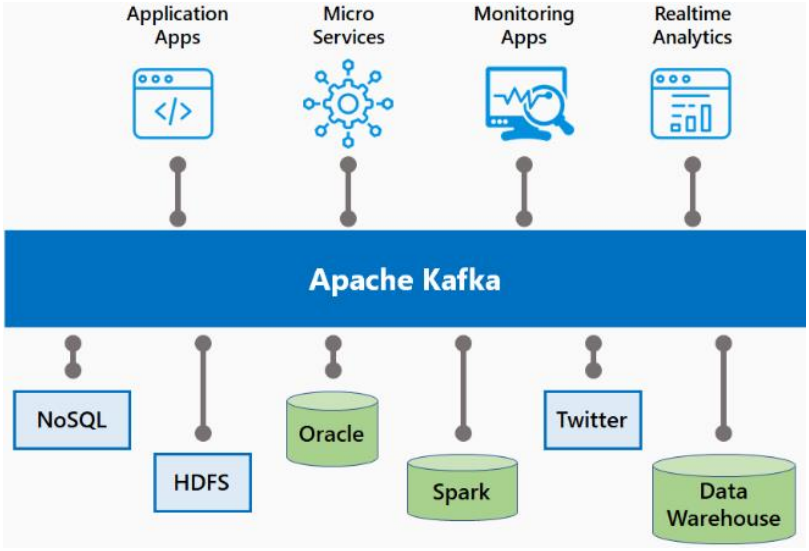

Рисунок 1.8 – Система обміну даними «Apache Kafka»

Ці платформи є розподіленими механізмами обробки, які дозволяють виконувати ланцюжки завдань обробки даних за шаблоном «видавець/абонент». Отримані потоки готові до використання як вхідних даних у наступних процесах або для збереження [12]. У сценарії вихідна швидкість системи має бути вищою, ніж швидкість потоку вхідних даних, щоб уникнути проблем із пам’яттю.

1.3.3 BigData-архітектури

Обсяг даних, які необхідно обробити, швидко збільшується, і з’являються нові вимоги до затримки та пропускної здатності. Офлайн- та онлайн-системи обробки даних інтегруються поступово, і сьогодні багато систем управління та обробки великих даних впроваджують обидва конвеєри паралельно (див. рисунок 1.9). У цьому контексті виникають дві архітектури обробки даних.

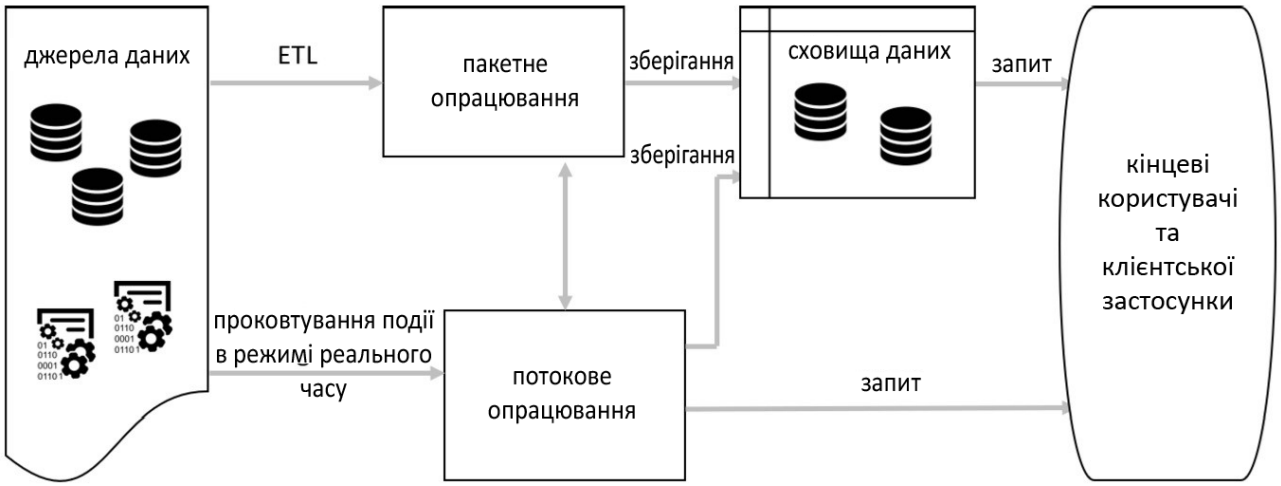

Рисунок 1.9 – Компоненти архітектури великих даних загального призначення

Архітектура «Лямбда» (див. рисуно 1.10) пропонує використовувати два конвеєри «Холодний шлях» та «Гарячий шлях» [29]. Дані потокової передачі надсилаються на обидва шляхи. «Гарячий шлях» характеризується високою пропускною здатністю, низькою затримкою, незначними помилками та аналітикою в режимі майже реального часу.

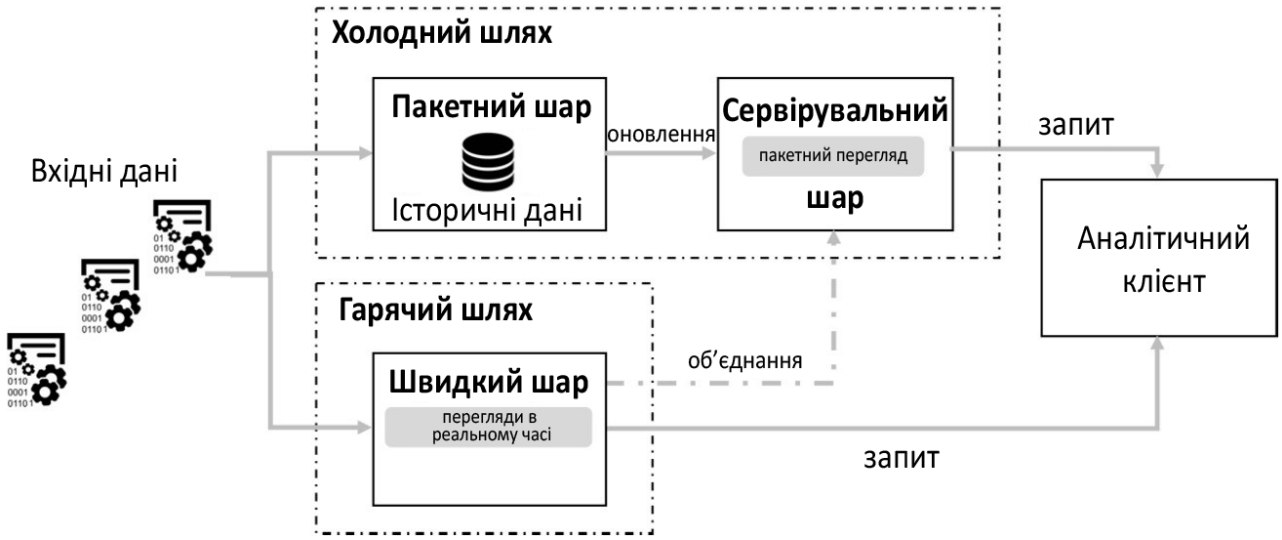

Рисунок 1.10 – Архітектура Лямбда (адаптовано з [29])

Потоки даних обробляються на рівні швидкості і слугують для перегляду в режимі реального часу. Цей шар враховує лише нові дані. «Холодний шлях» об'єднує вхідні дані з історичними даними і створює збагачені та детальні дані. «Холодний шлях» має два шари. Перший, пакетний рівень, поєднує потокові дані з історичними даними, а другий, рівень обслуговування, обслуговує дані, згенеровані на попередньому етапі – індексовані пакетні перегляди. Дані доступні не так швидко, як на «Гарячому шляху», але дані очищаються та готуються до використання ззовні [12]. Відповіді на запити створені на основі переглядів у режимі реального часу та індексованих групових переглядів.

Архітектура «Каппа» (див. рисунок 1.11) має лише один шлях, тобто потокова і пакетна обробка відбуваються по одній траєкторії [29].

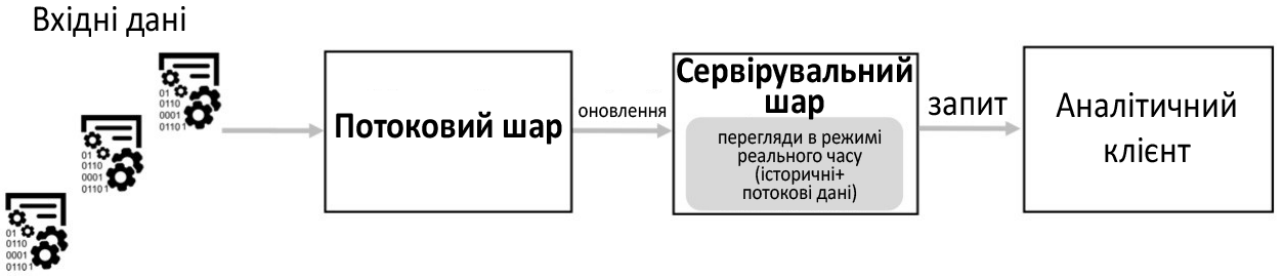

Рисунок 1.11 – Архітектура «Каппа» (адаптовано з [29])

Цей шлях також має два шари. Потоковий рівень обробляє вхідні дані, а обслуговуючий рівень обробляє дані, підготовлені на попередньому етапі, доступними для запиту. Ці дані також можуть бути подані в допоміжне сховище для обслуговування аналітичних клієнтів. Описані архітектури мають однакову мету, яка полягає в аналітичній обробці в режимі реального часу та історичної аналітики в єдиному середовищі. Однак архітектура «Каппа» обробляє обидва типи робочого навантаження, використовуючи один технологічний стек, тоді як архітектура «Лямбда» використовує технологічний стек для пакетної обробки [12].

Впродовж тривалого періоду часу були окремі системи «OLTP» і «OLAP». Отже, існує затримка між тим, що може бачити система «OLTP», і тим, що може запитувати система «OLAP». Цей підхід відповідає вимогам багатьох застосунків, але на даний час присутній інтерес до інтеграції систем «OLTP» і «OLAP» в єдину систему [12]. Це означає перехід від традиційного підходу, заснованого на сховищах даних і ETL, або на потоках подій, до архітектури, сформованої на озерах даних, де дані з усіх доменів, зокрема, транзакційних, аналітичних і потокових, доступні за допомогою конекторів – посередників і розподілених запитів (див. рисунок 1.12).

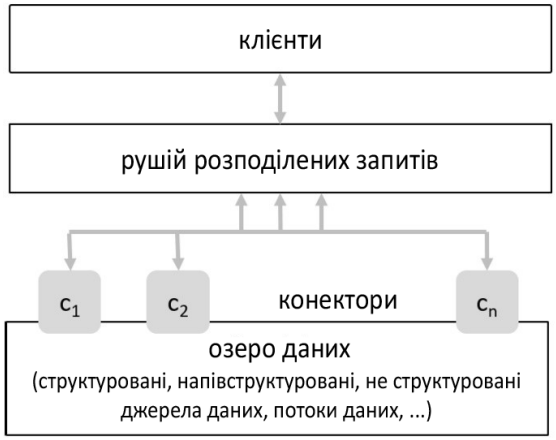

Рисунок 1.12 – Розподілені запити до гетерогенних баз даних

Це означає, що механізм запитів може отримувати дані з озера даних, перетворювати їх і зберігати результати назад в озері даних.

1.4 Бенчмаркінг великих даних

Порівняльний аналіз включає оцінку показників продуктивності при виконанні стандартизованих робочих навантажень з використанням різних систем над одним набором даних [12]. Системи великих даних включають різноманітні моделі представлення даних, архітектури обробки та розподілені схеми завдяки значній швидкості даних та характеристикам різноманітності. Це робить порівняльний аналіз систем великих даних дещо складнішим, ніж порівняльний аналіз традиційних (реляційних) систем баз даних.

Контрольні показники мають відмінні особливості [12]. З точки зору операцій:

– SQL-оператори;

– алгоритми обходу та аналізу графа;

– стискання та підрахунок електронних повідомлень;

– кластеризація та класифікація.

З позиції процесу ґенерації даних:

– чисті синтетичні ґенератори;

– ґенератори даних на основі реального світу;

– гібридні генератори даних.

В контексті систем великих даних, можемо згрупувати загальнодоступні показники за категоріями:

– показники продуктивності;

– показники ціни та ефективності;

– показники споживання енергії [30].

Системи порівняння також мають відмінні особливості та цілі з позиції оцінки показників продуктивності:

– пропускна здатність;

– співвідношення доступного часу системи та інтенсивності обчислень процесора.

1.5 Огляд налаштування систем БД

Продуктивність має вирішальне значення для систем БД. Це головна мета оптимізації налаштувань, і на неї впливає декілька аспектів, наприклад, кількість даних або вимоги до реплікації та узгодженості [31].

Налаштування продуктивності БД – це процес зміни та налаштувань на рівні БД, які покращать продуктивність програм БД.

Процес налаштування БД може здійснюватися безперервно в циклі зворотного зв’язку (див. рисунок 1.13). Виконуючи операції:

– відстежувати поведінку системи;

– складати плани можливих дій з налаштування;

– оцінювати та виконувати їх у БД;

– знову контролює систему БД, щоб спостерігати за впливом на робоче навантаження.

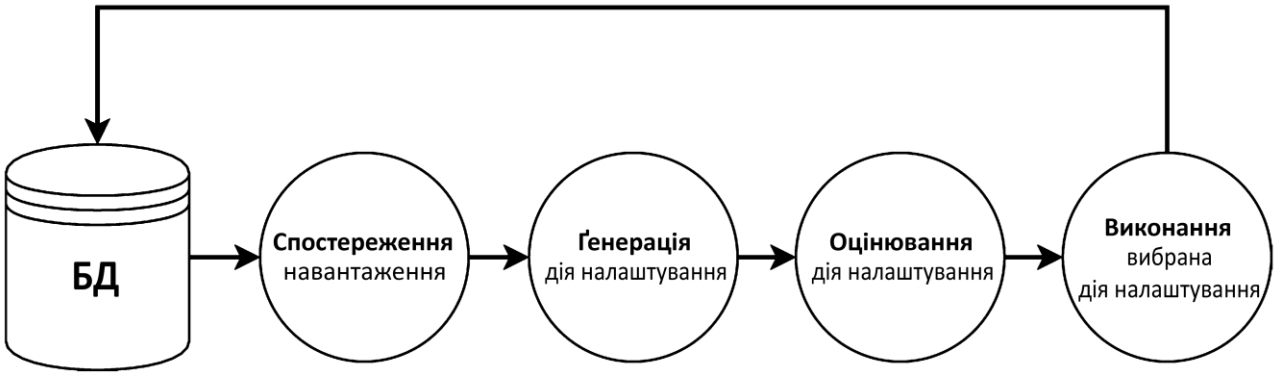

Рисунок 1.13 – Налаштування БД

Хоча більшість веб-даних є неструктурованими, компанії все ще враховують транзакційні дані, що зберігаються в RDBM, у своїх бізнес-рішеннях. Системи класу «HadoopDB» [32], дозволяють «MapReduce» через розподілені системи керування реляційними базами даних, наприклад, «PostgreSQL» і «MySQL».

1.6 Висновок до першого розділу

В першому розділі кваліфікаційної роботи подано означення великих за обсягом даних. Описано великі дані та аналітику великих даних. Розглянуто типові архітектури для аналітики великих даних. Прокласифіковано типи BigDate-систем. Описано типи BigDate-навантаження. Розглянуто BigDate-архітектури. Висвітлено поняття бенчмаркінґу великих даних. Подано огляд налаштування систем баз даних.

РОЗДІЛ 2. АНАЛІЗ МЕТОДІВ ПІДВИЩЕННЯ ПРОДУКТИВНОСТІ ЗАСОБІВ ОПРАЦЮВАННЯ ВЕЛИКИХ ДАНИХ

2.1 Підходи до підвищення продуктивності засобів опрацювання великих даних

Можна класифікувати основні дії щодо налаштування продуктивності на дві категорії [12]:

– *Рівень спостереження*, який є фундаментальним кроком, який не повинен бути шкідливим для доступності та виконання системи.

– *Дії налаштування самі по собі*, починаючи від звичайних фізичних порушень, таких як створення індексів і матеріалізованих представлень, до останніх і нових підходів, заснованих на машинному навчанні.

На рисунку 2.1 представлена класифікація основних дій налаштування, орієнтовані на дані [12].

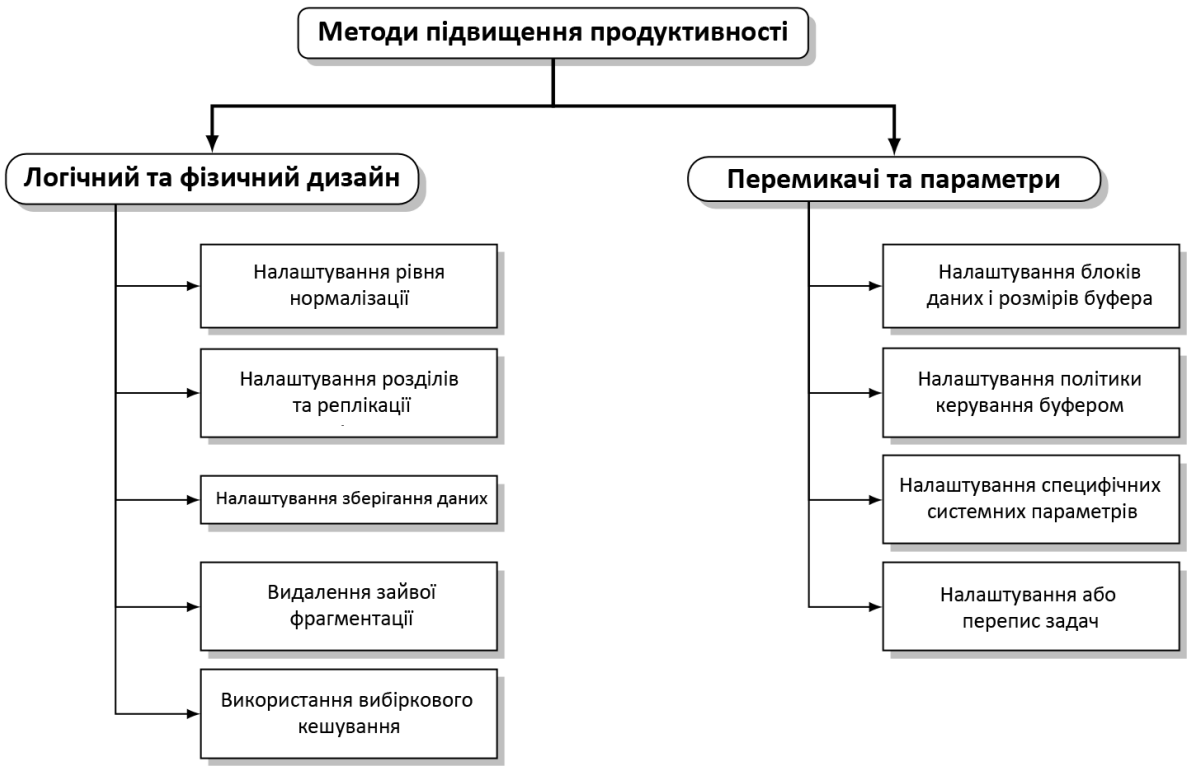

Рисунок 2.1 – Класифікація основних підходів до орієнтованого на дані налаштування

2.1.1 Логічне та фізичне оформлення

Методи кешування передбачають створення постійних структур зберігання:

– матеріалізовані уявлення;
– динамічне виділення доступної пам’яті на основі її призначення;
– кеш буфера даних;
– кеш бібліотеки тощо.

Матеріалізовані уявлення зберігають попередньо обчислені об’єднані та відфільтровані, агреговані та відсортовані дані. Вони також обмежують реляційні таблиці по вертикалі, кешуючи на диску лише стовпці, необхідні для запитів. Матеріалізоване уявлення може прискорити час відповіді бази даних, зменшуючи вартість запитів до бази даних, споживаючи менше ресурсів ЦП, пам’яті та вводу-виводу [33]. Однак потрібно враховувати використання дискового простору.

Індекси – це допоміжні структури, які забезпечують швидкий доступ до даних на основі одного або декількох ключових значень. Можна визначити проблему вибору індексу (ISP) як ситуацію, коли потрібно ідентифікувати та створити відповідні індекси для даного робочого навантаження, що супроводжується мінімізацією часу відповіді з дотриманням обмежень загального простору для зберігання індексів.

Це NP-складна проблема, враховуючи різноманітність таблиць, до яких звертаються команди SQL у робочому навантаженні, та різні типи індексів, зокрема:

– B-дерево;
– растрове зображення;
– текст;
– хеш;
– часткові;
– кластеризовані;

– геопросторові.

Оскільки наявність більшої кількості індексів для атрибутів, які часто використовуються в операціях запису, може збільшити навантаження на систему та оскільки для кожного оновлення також повинні оновлюватись структуру індексу [34] то окрема проблема полягає в досягненні компромісу між операціями читання та запису. Підходи до індексних рекомендацій, реалізовані RDBM, використовують оптимізатор бази даних і зібрану статистику для оцінки вартості запитів на основі гіпотетичних індексів [12].

2.1.2 Перемикачі та параметри систем

Однією з перших головних проблем DBA є те, як налаштувати системні ресурси для обмеження ресурсів з точки зору:

– обсягу пам’яті;

– розмірів кешу;

– типів дисків;

– параметрів операційної системи.

З точки зору параметрів, що стосуються пам’яті СУБД, добре виконана конфігурація розміру буфера та його політики може підвищити дискові процедури вводу-виводу та ефективність пам’яті [12].

Буферний пул містить у пам’яті зображення сторінок бази даних на диску, які на використовуються поточний момент часу, і зберігає ті, які незабаром будуть використані знову. Менеджер буфера – це компонент системи керування БД, відповідальний за заміну сторінок у відповідь на збої буфера та політику збереження пулу буферів. Його мета – мінімізувати доступ до жорсткого диска та забезпечити оптимальне використання основної пам’яті [35] зі збільшенням коефіцієнта звертання.

Багато робіт [12] з налаштування на рівні сторінки зосереджені на стратегіях заміни сторінок. Вони були проведені для оптимізації класичних алгоритмів управління буфером, орієнтованих на жорсткий диск, зокрема LRU,

MRU або MFU. Були запропоновані нові підходи з використанням імовірнісних адаптивних алгоритмів у поєднанні з нещодавньою появою дисків на основі флеш-пам’яті [36].

Налаштування параметрів конфігурації є складною діяльністю, оскільки більшість систем БД пропонують багато регуляторів налаштування, і деякі з них можуть по-різному впливати на продуктивність різних типів робочих навантажень, наприклад, OLTP, OLAP та гібридні налаштування. У розподіленому середовищі складність збільшується, оскільки системним адміністраторам також доводиться налаштовувати параметри конфігурації з вузлів зберігання – ЦП, сховище, пам’ять, і його комунікаційної мережі [12].

2.1.3 Техніки машинного навчання для підвищення продуктивності BigData

Дослідження щодо автоматичного налаштування БД, застосовують машинне навчання для налаштування регуляторів бази даних [12]. Для цього вони використовують методи:

– навчання під наглядом;
– навчання без нагляду;
– навчання з підкріпленням;
– нейронні мережі;
– методи глибокого навчання.

Зокрема, налаштування перемикачів є складною проблемою NP [12], тому, що має отримати користь від цих підходів, оскільки перемикачі здебільшого є числовими значеннями:

– категоріальними;
– дискретними;
– безперервними.

Вони використовуються для керування об’ємом кеш-пам’яті щодо того, як часто дані записуються в сховище та відображаються як функції. Вони

корисні на етапі навчання моделі та для прогнозування розрахункових значень параметрів налаштування [37]. Автоматично отримані знання щодо налаштування БД, можна повторно використати, щоб зменшити кількість часу та ресурсів, що використовуються для налаштування СУБД для конкретного робочого навантаження або навіть для нової програми.

У роботі [38] ці методи використовують для підтримки грубозернистого налаштування, наприклад, налаштування на рівні робочого навантаження. У [39] подана архітектура, яка постійно збирає, ненав'язливим способом, дані про робоче навантаження, поточні параметри та показники продуктивності СУБД і навчає цими даними нейронну мережу. Таким чином, можна знайти зв'язки між параметрами-кандидатами, які підлягають корекції, та показниками продуктивності, подати їх у алгоритм самонастроювання, який може налаштовувати одночасно декілька параметрів.

«OtterTuning» [38] – це інструмент, який використовує машинне навчання, контрольовані та неконтрольовані методи, щоб тренувати модель шляхом повторного використання даних навчання, зібраних під час попередніх сеансів налаштування, для вибору найбільш ефективних регуляторів та рекомендації найбільш підходящих налаштувань регуляторів у різних робочих навантаженнях СКБД. Він використовує лінійну регресію та статистичний метод, щоб визначити міцність зв'язку між однією або декількома залежними змінними та кожною з незалежних змінних.

2.2 Налаштування великомасштабних систем обробки BigData

Великомасштабні системи обробки BigData мають сотні параметрів, що налаштовуються. Деякі з них можуть суттєво впливати на продуктивність [40]. Наявність «розумного» набору параметрів, які можуть допомогти налаштувати виконання програми до певного контексту, може здатися бажаним, але налаштування занадто великої кількості параметрів для досягнення найкращої

конфігурації з точки зору пропускної здатності часто виявляється складним і трудомістким завданням [41].

Геродоуа [42] визначив три категорії можливостей оптимізації для аналітичних навантажень великих даних:

– спільне використання потоків даних (тобто одна робота виконує обчислення для різних логічних вузлів);

– ефективне використання матеріалізації проміжних результатів

– автоматична реорганізація проміжного зберігання даних за допомогою нових макетів даних, наприклад, з використанням розділення, та механізмів зберігання моделей, наприклад, на основі стовпців, які можуть краще підходити для розглянутого робочого процесу.

Отже, потрібно переглянути фактичне поняття налаштування системи БД. Однак на даний час не існує поняття оптимізатора, який може реагувати на нові структури даних і методи доступу [12].

2.3 Розміщення сховища та розміщення даних

У традиційних системах БД операції вводу-виводу зазвичай є найдорожчими. Зменшення потреби в операціях вводу-виводу або підвищення продуктивності вводу-виводу може істотно вплинути на загальну продуктивність системи. У цьому контексті SSD-диски використовувалися впродовж декількох останніх років, демонструючи значне покращення продуктивності, коли вони використовувалися замість магнітного диска для файлів із послідовним доступом, зокрема для журналів транзакцій та сегментів відкату [43].

В роботі [44] розглянуто використання SSD-дисків для підтримки систем великих даних. Особливу увагу було приділено впливу SSD-дисків на продуктивність «Hadoop». Хоча твердотільні накопичувачі мають кращу продуктивність порівняно з традиційними магнітними сховищами, при заміні магнітного сховища на SSD диски слід враховувати схеми доступу та обробки

даних [44]. В реальних сценаріях високої продуктивності жорсткі диски використовуються для постійного зберігання даних, оскільки вони більш економічно ефективні, ніж SSD [4].

У [45] продуктивність системи на основі «MapReduce» покращена за допомогою SSD, що використовується для зберігання проміжних даних, тобто вихідних даних карти, які тимчасово зберігаються локально. Стиснення проміжних даних також може підвищити продуктивність систем вводу-виводу на основі «MapReduce», але це не впливає істотно на продуктивність HDFS [46].

В роботі [47] автори пропонують систему управління даними, яка прогнозує шаблони доступу до файлів і використовує SSD для зберігання кешованих даних у HDFS. Автори стверджують, що загальну пропускну здатність вводу-виводу можна збільшити, якщо використовувати SSD для кешування деяких даних замість зберігання всіх даних.

2.4 Тиражування даних, передача даних, відмовостійкість та продуктивність BigData

У розподілених системах БД можна використовувати реплікацію даних для підвищення доступності та відмовостійкості. Разом з розділенням, реплікація даних відіграє вирішальну роль у продуктивності розподілених систем. В таблиці 2.1 подано деякі проблеми, пов’язані з підтримкою скороченого вводу/виводу [48].

Хоча користувачі часто можуть вказати користувацькі функції розділення, дізнатися розподіл даних може бути дещо складно. Потрібно розділити дані, які є результатом відображення, а не самі вхідні дані [48]. Крім того, передача даних між вузлами може суттєво вплинути на загальну продуктивність [4].

Таблиця 2.1 – Складні випадки для підтримки низького рівня вводу-виводу в «MapReduce»

Перекос розміру запису	**Перекос розбиття**
Поширені в напівструктурованих і неструктурованих системах даних	Призводить до розбалансування робочого навантаження між вузлами
Обчислювальний перекос	**Неоднорідність продуктивності**
Деякі завдання займають набагато більше часу, ніж інші.	Навіть між ідентичними вузлами кластера

У великомасштабних кластерах збої є доволі поширеними [48] відмовостійкість потрібна, щоб уникнути витрат на повне повторне виконання великої кількості тривалих завдань. У реалізаціях «MapReduce» витрати на відновлення збоїв на рівні завдання зазвичай зменшуються за рахунок використання постійних проміжних файлів.

2.5 Кешування та налаштування пам’яті

Кешування основної пам’яті може значно покращити роботу, пов’язану з БД. Автори [49]оцінюють конфігурації кешування в пам’яті та пропонують автоматичний вибір стратегії кешування для використання в «Spark» [50]. Але при цьому стверджують, що їхні пропозиції можуть застосовуватися до пам’яті для інших розподілених платформ. Запропонований підхід враховує:

– розмір файлу у файловій системі;

– доступну кластерну основну пам’ять;

– накладні фактори, наприклад, через серіалізацію та стиснення, пов’язані зі стратегіями кешування.

Опубліковані результати показують покращення продуктивності порівняно з налаштуваннями за замовчуванням. Налаштування параметрів, пов’язаних з пам’яттю, обговорюється в [51].

У висококонкурентних системах кожне завдання отримує лише невеликий шматок робочої пам’яті, а проміжні дані та результати можуть бути перенесені на диск. Таке завантаження диска може статися в реляційній СУБД, яка виконує занадто велику кількість одночасних SQL-запитів з операціями сортування, наприклад, як частина об’єднань та агрегацій, з аналітикою на основі «MapReduce», яка одночасно виконує занадто багато завдань [4]. Потім, щоб уникнути погіршення продуктивності через завантаження диска в «MapReduce», кожен вхідний розмір завдання повинен відповідати доступній пам’яті [4]. Слід зауважити, що зменшення кількості завдань може збільшити розмір даних, які споживає кожне з них, і має існувати порогове значення, зазвичай визначене експериментально, яке може призвести до зменшення навантаження на диск.

2.6 Сховища великих даних і логічна організація

Великі дані і традиційні сховища не є несумісними, і їх об’єднання може призвести до відповідних переваг [52]. Хоча більшість поточних робіт щодо сховищ великих даних зосереджені на використанні сховищ «MapReduce» і «NoSQL» [52], присутні дослідницькі роботи, які поєднують використання вже встановлених аналітичних можливостей традиційних сховищ даних із технологіями великих даних [53].

Організація даних зі схемою зірка є поширеним підходом у традиційних сховищах. Коста [53] використовує схему «зірка» в контексті великих даних і представляє оцінку продуктивності при виконанні запиту над розділеними даними. Автори порівнюють використання зіркової схеми з повністю денормалізованою таблицею та відзначають, що вона має тенденцію перевершувати першу за продуктивністю.

В [54] автори обговорюють пакетні та потокові аналітичні робочі навантаження в контексті сховищ великих даних. Вони описують тест «SSB+», сформований на «TPC-H» [55]. Він містить нові структури даних і ґенератори

для представлення потокових даних і робочих навантажень – тобто моделювання даних соціальних мереж, та опису результатів продуктивності за допомогою сховища NoSQL-даних «Cassandra», «Hive» і «Presto» з «SSB+».

Що стосується логічного розташування даних, автори порівнюють використання плоских таблиць та зіркової схеми в сховищах великих даних і приходять до висновку, що розмір вимірів може суттєво вплинути на те яку структуру сховища слід використовувати [12]. Вони пропонують, щоб спеціалісти-практики виконували попередній аналіз, використовуючи вибіркові дані, щоб вибрати між схемою «зірка» та плоскими таблицями.

2.7 Рекомендації для підвищення продуктивності засобів аналітичного опрацювання великих даних

Перший крок у налаштуванні продуктивності зазвичай полягає в тому, щоб визначити, які компоненти потрібно налаштувати, і які можливі зміни необхідно внести. Найпоширеніші зміни налаштувань БД пропонують певні компромісні рішення [12]. Точний вплив невеликої зміни в компонент або параметр системи на всю систему заздалегідь невідомий. Наприклад, вибір розміру блоку даних є компромісним рішенням між активністю заміни сторінки та ступенем паралельності. Використання стиснення даних можна вважати ще одним компромісним рішенням, оскільки воно зменшує витрати на ввід-вивід, але використання сильно стиснутих даних збільшує витрати ЦП. Тому, перед запуском необхідно ретельно оцінити можливі дії щодо налаштування [12].

Дії налаштування можуть не мати простої реалізації, наприклад, розділення даних у системах на основі «MapReduce». У цьому контексті розділ вхідних даних слід виконувати з врахуванням заздалегідь невідомих результатів відображення. Погано розроблена схема розподілу даних може не тільки знизити ефективність розподілу, але й призвести до деяких накладних витрат на обробку залежно від типу створеного перекосу.

Налаштування також може передбачати компроміси з точки зору витрат на відновлення системи та продуктивності. Наприклад, матеріалізація проміжних результатів може покращити відновлення після збою, але також погіршити продуктивність виконання завдання [12]. Інший компроміс стосується продуктивності та фінансових витрат. Високопродуктивне сховище демонструє значне покращення продуктивності порівняно з магнітними дисками, але зберігання всіх даних на SSD може мати непомірні витрати.

Тим не менш, все ще можна перерахувати дії з налаштування, які, ймовірно, стануть передовою практикою, яку потрібно оцінити під час підвищення продуктивності системи, узагальнюючи більшість проаналізованих в кваліфікаційній роботі публікацій. Вибрані рекомендації організовано відповідно до середовища, де вони найімовірніше будуть ефективними. Однак ми все одно можемо застосовувати еквівалентні рекомендації в інших налаштуваннях [12]. Загальні рекомендації:

– При заміні магнітного накопичувача на твердотільний потрібно враховувати схеми доступу та обробки даних [44].

– Для ефективного розподілу даних потрібно співставляти характеристики робочого навантаження в атрибути файлової системи [46].

– Під час налаштування пристроїв зберігання даних потрібно враховувати кількість і типи пристроїв [46].

Рекомендації для систем на основі «MapReduce» [12]:

– Для зберігання проміжних даних доцільно використовувати SSD [45] щоб розмістити вихідні дані карти, які тимчасово зберігаються локально.

– Доцільно стиснути проміжні дані [46].

– При використанні малих розмірів блоків потрібно оцінити збільшення кількості потоків [41].

– Потрібно уникайте використання великих розмірів блоків, оскільки вони можуть вплинути на паралельність [41].

– Потрібно кешувати дані розділу, які є результатом відображення, а не самі вхідні дані [48].

– Слід уникати передачі занадто великої кількості даних між вузлами [4].

– У кластерах, де збої трапляються рідко, можна зменшити матеріалізацію проміжних даних, щоб підвищити продуктивність [48].

– Щоб виконати сортування в пам’яті доцільно використовувати невеликі розділи карт [48].

– Щоб оцінити приблизний розподіл отриманих на карті даних слід використовувати вибірку [48].

– Слід прийняти виважене рішення, чи слід сортувати вихідні дані редукторів, щоб вибрати політику розділу. [48].

– Злиття фраґментованих проміжних файлів у більші блоки, з великими послідовними запитами вводу-виводу, щоб впоратися з перекосами в проміжних даних [4].

– Щоб уникнути перевантаження диска кожен вхідний розмір завдання повинен відповідати доступній пам’яті [4].

– Щоб зменшити перевантаження диска потрібно налаштувати кількість завдань і розмір даних, які споживає кожне з них [4].

Рекомендації для систем на основі «HDFS» та «Hive» [12]:

– Кешовані дані доцільно розмістити на SSD [47].

– Багаторівневе розділення слід використовувати обережно, щоб уникнути надмірного розподілу, що призводить до зростання кількості рівнів папок з невеликими файлами та меншою кількістю даних на папку [53].

– Доцільно використовувати групування одночасно в двох таблицях за атрибутом, який використовується для їх об’єднання. [53].

Рекомендації для систем на основі «Spark» [12]:

– Щоб вибрати стратегію кешування потрібно перевірити розмір файлу, доступну загальнокластерну основну пам’ять і накладні витрати на представлення розподілених об’єктів, [49].

– Доцільно уникати стиснення та серіалізації, якщо головна пам'ять кластера може кешувати весь набір даних [49].

2.8 Висновок до другого розділу

В другому розділі кваліфікаційної роботи проаналізовано підходи до підвищення продуктивності засобів опрацювання великих даних. Зокрема розглянуто логічне та фізичне оформлення великих даних. Описано перемикачі та параметри систем. Висвітлено техніки машинного навчання для підвищення продуктивності BigData. Проаналізовано налаштування великомасштабних систем обробки BigData. Розглянуто розміщення сховища та розміщення даних. Описано тиражування даних, передача даних, відмовостійкість та продуктивність BigData. Висвітлено кешування та налаштування пам’яті. Розглянуто сховища великих даних і логічну організацію. На основі обширного переліку наукових публікацій запропоновано рекомендації для підвищення продуктивності засобів опрацювання великих даних по категоріях.

РОЗДІЛ 3. БЕЗПЕКА ЖИТТЄДІЯЛЬНОСТІ, ОСНОВИ ОХОРОНИ ПРАЦІ

3.1 Управління та нагляд за безпекою життєдіяльності в Україні

В кваліфікаційній роботі проведено аналіз методів підвищення продуктивності засобів опрацювання великих даних. На даний час інформаційно-технологічні платформи з використанням великих даних використовуються в багатьох сферах людської діяльності та галузях виробництва. Тому доцільно розглянути питання управління та нагляду за безпекою життєдіяльності в Україні.

Основою управління безпекою є система організаційно-розпорядчих заходів з профілактики і протидії негативним факторам – недоліки, проблеми, кризові ситуації, які порушують стійке функціонування та розвиток держави, регіону, погрожуючи небезпекою окремій людині [56].

Управління – це завжди система взаємопов'язаних елементів. Вона складається з чотирьох основних структур. Перша – суб'єкти управління – органи, що відповідають за управління безпекою. Друга – об'єктивна система принципів, правил, певних обмежень, які формують структуру управління. Третя - сукупність інформаційних процесів, необхідних для регулювання стану об'єкта, його контроля. Четверта – кваліфікований, структурований персонал, здатний контролювати ситуацію, приймати рішення. Ця структура багатопланова, складна. Її потрібно розглядати на загальнодержавному, регіональному, місцевому та локальному рівнях.

Вся державна система – Кабінет Міністрів України, Національна Рада з питань безпеки життєдіяльності населення, Комітет з нагляду за охороною праці, структури Міністерства безпечної життєдіяльності, Служба безпеки України, Міністерство внутрішніх справ, органи державного пожежного нагляду, місцеві державні адміністрації та Ради депутатів, їх Виконавчі комітети.

Важливим державним органом є Національна рада з питань безпечної життєдіяльності населення, яка створена відповідно до Закону України «Про охорону праці». Основне її призначення розробка та реалізація державної політики в галузі охорони життя людей на виробництві та профілактики побутового травматизму, створення системи державного управління цією галуззю.

Національна рада у своїй діяльності керується Конституцією і законами України, постановами Верховної Ради України, указами і розпорядженнями Президента України, декретами, постановами і розпорядженнями Кабінету Міністрів України, а також Положенням про Національну раду з питань безпеки. Вона розробляє та здійснює заходи щодо створення цілісної системи державного управління охороною життя людей на виробництві та профілактики побутового травматизму, вносить на розгляд Кабінету Міністрів України пропозиції про вдосконалення цієї системи; організує і забезпечує контроль за виконанням законодавчих актів і рішень Уряду України.

Рада сприяє впровадженню в життя Національної програми з безпечної життєдіяльності та законів, пов'язаних з реалізацією державної політики з питань безпечної життєдіяльності населення. Вона подає Кабінету Міністрів України пропозиції щодо вдосконалення законодавства з цих питань та координує діяльність центральних і місцевих органів державної виконавчої влади в галузі охорони життя людей на виробництві та профілактики побутового травматизму [57].

Цей орган організує перевірки діяльності центральних і місцевих органів державної виконавчої влади і заслуховує на своїх засіданнях або засіданнях бюро Національної ради звіти керівників з питань, що входять до її компетенції. Її представники беруть участь у міжнародному співробітництві, сприяючи вивченню, узагальненню і поширенню досвіду у галузі охорони життя людей на виробництві та профілактики побутового травматизму, вирішує питання контролю за виконанням укладених договорів і угод у цій галузі.

Рішення Національної ради та її бюро, прийняті в межах їх компетенції, є обов'язковими для виконання центральними та місцевими органами державної виконавчої влади, підприємствами, установами, організаціями та громадянами.

Крім центральних державних органів управління та нагляду за станом безпеки важливе місце займають місцеві органи – обласні, міські, на виробничих об'єктах, які повинні контролювати стан безпеки, охорони праці на кожному підприємстві [58].

Важливу роботу виконують державні інспектори, які проводять обстеження стану підприємств, фіксують порушення нормативних актів з охорони праці, призупиняють роботу виробництв та об'єктів, на яких виникала пряма загроза здоров'ю, життю працюючих. Велику роль відіграють експертно-технічні центри, які займаються технічною експертизою стану обладнання, промислових об'єктів.

Важливе значення має робота з перегляду нормативно правових актів щодо державної експертизи проектної документації з питань охорони праці, здоров'я робітників, експертизи екологічного стану підприємств, місць їх розташування.

3.2 Контроль за станом охорони праці

Кваліфікаційна робота освітнього рівня «бакалавр» присв’ячена аналізу методів підвищення продуктивності засобів опрацювання великих даних. Інформаційно-технологічні платформи з використанням великих даних використовуються в обширному переліку сфер людської діяльності. Тому доцільно розглянути питання контролю за станом охорони праці.

Контроль за станом охорони праці на підприємстві є одним із складових елементів системи управління охороною праці підприємства. Контроль спрямований на виявлення відхилень від норм в умовах праці та перевірку виконання працівниками своїх обов’язків у сфері охорони праці. Дізнайтеся, які бувають основні види контролю за станом охорони праці на підприємстві, з

якою періодичністю їх проводять, та якими документами супроводжується контроль.

Контроль за станом охорони праці (далі – контроль) є найбільш відповідальною та трудомісткою функцією процесу управління, від якої залежить система управління охороною праці підприємства в цілому [59]. Оперативно виявити можливі відхилення від норм безпеки праці, перевірити виконання запланованих заходів та управлінських рішень можливо лише на підставі регулярного та об'єктивного контролю на підприємстві. Контроль має здійснюватися керівниками всіх рівнів управління виробництвом. При створенні безпечних умов праці на підприємстві значну роль також відіграє громадський контроль, що провадиться громадськими інспекторами, представниками профспілок або уповноваженими особами з питань охорони праці у разі відсутності профспілки).

Організація контролю за станом охорони праці на підприємстві тісно пов'язана із управлінням підприємством. Систему управління підприємством можна умовно поділити на рівні за певною ознакою, для кожного з яких має бути відповідний ступінь контролю.

Відповідно до масштабу підприємства розрізняють п'ять рівнів системи управління підприємством. Кількість рівнів залежить від розгалуженості структури підприємства:

– однорівнева система – у суб'єктів малого підприємництва і приватних підприємців;

– дворівнева система – на малих підприємствах – приватні підприємства;

– трирівнева система – на великих підприємствах – державні підприємства, акціонерні товариства;

– чотирирівнева система – на підприємствах із розвиненим корпоративним управлінням – дочірні підприємства, відомчі об'єднання підприємств;

– п'ятирівнева система – на підприємствах з розвиненим транснаціональним корпоративним управлінням, характерно для міжнародного

співробітництва міністерств, транснаціональних об’єднань, компаній, холдингів).

Залежно від системи управління підприємством та масштабів його підрозділів контроль за станом охорони праці може нараховувати від одного до п’яти ступенів [60]. Наприклад, на підприємствах електроенергетичного комплексу найпоширенішим є триступеневий контроль, а на підприємствах нафтогазового комплексу – п’ятиступеневий контроль за станом охорони праці.

Контроль за станом охорони праці на підприємстві або відокремленому структурному підрозділі можна поділити на:

– оперативний контроль за станом охорони праці (або поточний);

– вибірковий (або цільовий);

– періодичний.

На практиці основними видами контролю за станом охорони праці, які провадяться на підприємствах, є, зокрема:

– триступеневий адміністративно-громадський контроль;

– оперативний контроль керівниками робіт та іншими відповідальними особами підприємства, наприклад, службою охорони праці;

– контроль вищою організацією, четвертий або п’ятий ступінь контролю;

– контроль місцевими органами влади, органами самоврядування;

– відомчий контроль – контроль вищими органами, наприклад, міністерством;

– громадський контроль;

– контроль органами державного нагляду.

Більшість із цих видів контролю врегульовані законодавством, в якому чітко вказано, хто здійснює контроль за станом охорони праці на підприємстві. Зокрема, контроль органами державного та відомчого нагляду, місцевими органами влади (органами самоврядування) та громадський контроль регламентуються відповідними нормативно-правовими актами.

ВИСНОВКИ

В першому розділі кваліфікаційної роботи освітнього рівня «Бакалавр»:

– Подано означення великих за обсягом даних.

– Описано великі дані та аналітику великих даних.

– Розглянуто типові архітектури для аналітики великих даних.

– Прокласифіковано типи BigDate-систем.

– Описано типи BigDate-навантаження.

– Розглянуто BigDate-архітектури.

– Висвітлено поняття бенчмаркінґу великих даних.

– Подано огляд налаштування систем баз даних.

В другому розділі кваліфікаційної роботи:

– Проаналізовано підходи до підвищення продуктивності засобів опрацювання великих даних. Зокрема, розглянуто логічне та фізичне оформлення великих даних. Описано перемикачі та параметри систем.

– Висвітлено техніки машинного навчання для підвищення продуктивності BigData.

– Проаналізовано налаштування великомасштабних систем обробки BigData.

– Розглянуто розміщення сховища та розміщення даних.

– Описано тиражування даних, передача даних, відмовостійкість та продуктивність BigData.

– Висвітлено кешування та налаштування пам’яті.

– Розглянуто сховища великих даних і логічну організацію.

– На основі аналізу обширного переліку наукових публікацій запропоновано рекомендації для підвищення продуктивності засобів опрацювання великих даних по категоріях.

У розділі «Безпека життєдіяльності, основи охорони праці» описано управління та нагляд за безпекою життєдіяльності в Україні. Висвітлено контроль за станом охорони праці.

ПЕРЕЛІК ДЖЕРЕЛ

1 D. Abadi, A. Ailamaki, D. Andersen, P. Bailis, M. Balazinska, P. Bernstein, P. Boncz, S. Chaudhuri, A. Cheung, A.H. Doan, L. Dong, M.J. Franklin, J. Freire, A. Halevy, J.M. Hellerstein, S. Idreos, D. Kossmann, T. Kraska, S. Krishnamurthy, V. Markl, S. Melnik, T. Milo, C. Mohan, T. Neumann, B.C. Ooi, F. Ozcan, J. Patel, A. Pavlo, R. Popa, R. Ramakrishnan, C. Ré, M. Stonebraker, D. Suciu, The Seattle report on database research, SIGMOD Rec. 48 (4) (2020) 44–53.

2 A. Arvanitis, S. Babu, E. Chu, A. Popescu, A. Simitsis, K. Wilkinson, Automated performance management for the big data stack, in: CIDR 2019 - 9th Biennial Conference on Innovative Data Systems Research, 2019.

3 Duda, O., et al, Selection of Effective Methods of Big Data Analytical Processing in Information Systems of Smart Cities. CEUR Workshop Proceedings 2631, pp. 68-78. 2020.

4 H. Zhang, B. Cho, E. Seyfe, A. Ching, M.J. Freedman, Riffle: optimized Shuffle service for large-scale data, in: Proceedings of the Thirteenth EuroSys Conference, 2018, pp. 1–15.

5 Duda, Oleksii, et al. "COVID-19 data collections and analytical processing." 2021 IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT). Vol. 2. IEEE, 2021.

6 Y. Riahi, S. Riahi, Big Data and Big Data analytics: concepts, types and technologies, Int. J. Res. Eng. 5 (9) (2018) 524–528.

7 E.G. Ularu, F.C. Puican, A. Apostu, M. Velicanu, Perspectives on Big Data and Big Data analytics, Database Syst. J. 3 (4) (2012) 3–14.

8 X. Jin, B.W. Wah, X. Cheng, Y. Wang, Significance and challenges of Big Data research, Big Data Res. 2 (2) (2015) 59–64.

9 U. Sivarajah, M.M. Kamal, Z. Irani, V. Weerakkody, Critical analysis of Big Data challenges and analytical methods, J. Bus. Res. 70 (2017) 263–286.

10 Duda, O., Pasichnyk, V., Kunanets, N., Antonii, R., & Matsiuk, O. (2020, September). Multidimensional Representation of COVID-19 Data Using OLAP

Information Technology. In 2020 IEEE 15th International Conference on Computer Sciences and Information Technologies (CSIT) (Vol. 2, pp. 277-280). IEEE.

11 D. Abadi, R. Agrawal, A. Ailamaki, M. Balazinska, P.A. Bernstein, M.J. Carey, S. Chaudhuri, J. Dean, A. Doan, M.J. Franklin, J. Gehrke, L.M. Haas, A.Y. Halevy, J.M. Hellerstein, Y.E. Ioannidis, H.V. Jagadish, D. Kossmann, S. Madden, S. Mehrotra, T. Milo, J.F. Naughton, R. Ramakrishnan, V. Markl, C. Olston, B.C. Ooi, C. Re, D. Suciu, M. Stonebraker, T. Walter, J. Widom, Beckman report on database research, Commun. ACM 59 (2) (2016) 92–99.

12 Costa, Rogério Luís de C., et al. "A survey on data-driven performance tuning for big data analytics platforms." Big Data Research 25 (2021): 100206.

13 Bodnarchuk I., Duda O., Kharchenko A., Kunanets N., Matsiuk O., Pasichnyk V. Choice method of analytical information-technology platform for projects associated to the smart city class. ICTERI 2020 ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer Proceedings of the 14th International Conference on ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer. Volume I: Main Conference p.317-330.

14 The Apache Software Foundation, Apache Hadoop, https://hadoop .apache .org/.

15 A. Thusoo, J.S. Sarma, N. Jain, Z. Shao, P. Chakka, S. Anthony, H. Liu, P. Wyckoff, R. Murthy, Hive: a warehousing solution over a map-reduce framework, Proc. VLDB Endow. 2 (2) (2009) 1626–1629.

16 M. Kornacker, A. Behm, V. Bittorf, T. Bobrovytsky, C. Ching, A. Choi, J. Erickson, M. Grund, D. Hecht, M. Jacobs, I. Joshi, L. Kuff, D. Kumar, A. Leblang, N. Li, I. Pandis, H. Robinson, D. Rorke, S. Rus, J. Russell, D. Tsirogiannis, S. Wanderman-Milne, M. Yoder, Impala: a modern, open-source SQL engine for Hadoop, in: CIDR 2015, Seventh Biennial Conference on Innovative Data Systems Research, Online Proceedings, Asilomar, CA, USA, January 4-7, 2015, 2015, pp. 1406–1415.

17 M. Armbrust, R.S. Xin, C. Lian, Y. Huai, D. Liu, J.K. Bradley, X. Meng, T. Kaftan, M.J. Franklin, A. Ghodsi, M. Zaharia, Spark SQL: relational data processing in spark, in: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, SIGMOD ’15, 2015, pp. 1383–1394.

18 Duda, O., Pasichnyk, V., Kunanets, N., Antonii, R., Matsiuk, O. Multidimensional Representation of COVID-19 Data Using OLAP Information Technology. International Scientific and Technical Conference on Computer Sciences and Information Technologies, 2020, 2, pp. 277–280, 9321889.

19 Apache Kudu - Fast Analytics on Fast Data, https://kudu .apache .org/.

20 A. Corbellini, C. Mateos, A. Zunino, D. Godoy, S. Schiaffino, Persisting big-data: the NoSQL landscape, Inf. Syst. 63 (2017) 1–23, https://doi .org /10 .1016 /j .is . 2016 .07.009.

21 B.G. Tudorica, C. Bucur, A comparison between several NoSQL databases with comments and notes, in: Proc. - RoEduNet IEEE Int. Conf., 2011.

22 R. Hecht, S. Jablonski, Nosql evaluation: a use case oriented survey, in: 2011 International Conference on Cloud and Service Computing, CSC, IEEE Computer Society, 2011, pp. 336–341.

23 E. Stefani, K. Hoxha, Implementing triple-stores using NoSQL databases, CEUR Workshop Proc. 2280 (2018) 86–92.

24 A.T. Kabakus, R. Kara, A performance evaluation of in-memory databases, J. King Saud Univ, Comput. Inf. Sci. 29 (4) (2017) 520–525, https:// doi .org /10 .1016 /j .jksuci .2016 .06 .007, http://www.sciencedirect .com /science / article /pii /S1319157816300453.

25 C. Li, B. Li, M.Z.A. Bhuiyan, L. Wang, J. Si, G. Wei, J. Li, Flutedb: an efficient and scalable in-memory time series database for sensor-cloud, J. Parallel Distrib. Comput. 122 (2018) 95–108, https://doi .org /10 .1016 /j .jpdc .2018 .07.021, http://www.sciencedirect .com /science /article /pii /S0743731518305422.

26 J. Arulraj, A. Pavlo, How to build a non-volatile memory database management system, in: S. Salihoglu, W. Zhou, R. Chirkova, J. Yang, D. Suciu

(Eds.), Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD Conference, ACM, 2017, pp. 1753–1758.

27 D. Kim, W.G. Choi, H. Sung, S. Park, A scalable and persistent key-value store using non-volatile memory, in: Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing, SAC ’19, Association for Computing Machinery, New York, NY, USA, 2019, pp. 464–467, https://doi .org /10 .1145 /3297280.3298991.

28 C. Barba-González, A.J. Nebro, A. Benítez-Hidalgo, J. García-Nieto, J.F. Aldana-Montes, On the design of a framework integrating an optimization engine with streaming technologies, Future Gener. Comput. Syst. 107 (2020) 538–550, https://doi .org /10 .1016 /j .future .2020 .02 .020, http://www. sciencedirect .com /science /article /pii /S0167739X19315699.

29 V. Persico, A. Pescapé, A. Picariello, G. Sperlí, Benchmarking big data architectures for social networks data processing using public cloud platforms, Future Gener. Comput. Syst. 89 (2018) 98–109.

30 R. Han, L.K. John, J. Zhan, Benchmarking Big Data systems: a review, IEEE Trans. Serv. Comput. 11 (3) (2018) 580–597.

31 S. Chaudhuri, G. Weikum, Foundations of automated database tuning, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Baltimore, Maryland, USA, June 14-16, 2005, ACM, Baltimore, Maryland, USA, 2005, pp. 964–965.

32 A. Abouzeid, K. Bajda-Pawlikowski, D. Abadi, A. Silberschatz, A. Rasin, Hadoopdb: an architectural hybrid of MapReduce and DBMS technologies for analytical workloads, Proc. VLDB Endow. 2 (1) (2009) 922–933, https:// doi .org /10 .14778 /1687627.1687731.

33 N. Noon, J. Getta, Automated performance tuning of data management systems with materializations and indices, J. Comput. Commun. 04 (2016) 47–53.

34 P. Ameri, On a self-tuning index recommendation approach for databases, in: 32nd IEEE International Conference on Data Engineering Workshops, ICDE Workshops, 2016, pp. 201–205.

35 R.A.P. Rangel, J.P. Ortega, J.A. Martínez Flores, J.J.G. Barbosa, Mirna P. Ponce F., Least likely to use: a new page replacement strategy for improving database management system response time, in: Computer Science - Theory and Applications, First International Computer Science Symposium in Russia, CSR, 2006, pp. 514–523.

36 A.O. Thakare, P.S. Deshpande, Probabilistic page replacement policy in buffer cache management for flash-based cloud databases, Comput. Inform. 38 (6) (2019) 1237–1271.

37 G. Li, X. Zhou, S. Li, B. Gao, Qtune: a query-aware database tuning system with deep reinforcement learning, Proc. VLDB Endow. 12 (12) (2019) 2118–2130.

38 D.V. Aken, A. Pavlo, G.J. Gordon, B. Zhang, Automatic database management system tuning through large-scale machine learning, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2017, pp. 1009–1024.

39 C. Zheng, Z. Ding, J. Hu, Self-tuning performance of database systems with neural network, in: Proceedings of 10th International Conference Intelligent Computing Theory ICIC, 2014, pp. 1–12.

40 J. Lu, Y. Chen, H. Herodotou, S. Babu, Speedup your analytics: automatic parameter tuning for databases and big data systems, Proc. VLDB Endow. 12 (12) (2018) 1970–1973.

41 G. Bansal, A. Gupta, U. Pyne, M. Singhal, S. Banerjee, A framework for performance analysis and tuning in Hadoop based clusters, in: Workshop on Smarter Planet and Big Data Analytics, SPBDA, 2014, pp. 1–6.

42 H. Herodotou, H. Lim, G. Luo, N. Borisov, L. Dong, B. Cetin, S. Babu, Starfish: a self-tuning system for Big Data analytics, in: CIDR 2011, Fifth Biennial Conference on Innovative Data Systems Research, 2011, pp. 261–272.

43 S.-W. Lee, B. Moon, C. Park, J.-M. Kim, S.-W. Kim, A case for flash memory SSD in enterprise database applications, in: Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data, 2008, pp. 1075–1086.

44 M. Bakratsas, P. Basaras, D. Katsaros, L. Tassiulas, Hadoop MapReduce performance on SSDs for analyzing social networks, Big Data Res. 11 (2018) 1–10.

45 S. Moon, J. Lee, X. Sun, Y. suk Kee, Optimizing the Hadoop MapReduce framework with high-performance storage devices, J. Supercomput. 71 (9) (2015) 3525–3548.

46 D.Q. Ren, B. Xia, File system performance tuning for standard Big Data benchmarks, in: Procs. Intl. Conf. on Computing and Data Engineering, ICCDE, 2018, pp. 22–26.

47 K.R. Krish, M.S. Iqbal, A.R. Butt, Venu: orchestrating SSDs in Hadoop storage, in: 2014 IEEE International Conference on Big Data (Big Data), 2014, pp. 207–212.

48 A. Rasmussen, V.T. Lam, M. Conley, G. Porter, R. Kapoor, A. Vahdat, Themis: an I/O-efficient MapReduce, in: Proceedings of the Third ACM Symposium on Cloud Computing - SoCC, ACM Press, 2012, pp. 1–14.

49 A.-K. Koliopoulos, P. Yiapanis, F. Tekiner, G. Nenadic, J. Keane, Towards automatic memory tuning for in-memory Big Data analytics in clusters, in: 2016 IEEE International Congress on Big Data (BigData Congress), 2016.

50 M. Zaharia, R.S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M.J. Franklin, A. Ghodsi, J. Gonzalez, S. Shenker, I. Stoica, Apache spark: a unified engine for Big Data processing, Commun. ACM 59 (11) (2016) 56–65.

51 K. Aziz, D. Zaidouni, M. Bellafkih, Leveraging resource management for efficient performance of Apache Spark, J. Big Data 6 (1) (2019) 78.

52 M. Pticekˇ , B. Vrdoljak, Big Data and new data Warehousing approaches, in: Proceedings of the 2017 International Conference on Cloud and Big Data Computing - ICCBDC 2017, ACM Press, 2017, pp. 6–10.

53 E. Costa, C. Costa, M.Y. Santos, Evaluating partitioning and bucketing strategies for hive-based Big Data Warehousing systems, J. Big Data 6 (1) (2019) 34.

54 C. Costa, M.Y. Santos, Evaluating several design patterns and trends in Big Data Warehousing systems, in: Advanced Information Systems Engineering, Springer International Publishing, 2018, pp. 459–473.

55 TPC Homepage, http://www.tpc .org.

56 Управління та нагляд за станом безпеки життєдіяльності. URL: https://pidru4niki.com/14360106/bzhd/upravlinnya_naglyad_stanom_bezpeki_zhittyediyalnosti.

57 Профілактика травматизму. URL: http://www.oht.sm.gov.ua/index.php/uk/gumanitarna-politika/sotsialniy-zahyst/6249-profilaktika-travmatizmu.

58 Організація охорони праці на підприємстві. URL: https://www.sop.com.ua/article/378-organzatsya-ohoroni-prats.

59 Контроль за станом охорони праці на підприємстві. URL: https://www.sop.com.ua/article/262-qqq-16-m1-11-01-2016-kontrol-za-stanom-ohoroni-prats-na-pdprimstv.

60 Як контролювати стан охорони праці на підприємстві: основні кроки у поміч. URL: https://nov-rada.gov.ua/2021/06/18/iak-kontroliuvaty-stan-okhorony-pratsi-na-pidpryiemstvi-osnovni-kroky-u-pomich/.