UDC 004.652.4:004.738.5
DOI https://doi.org/10.32689/maup.it.2025.2.27

Khrystyna TERLETSKA
Bachelor's Degree in "Documentation and Information Services",
Senior Software Engineer at Datadog,
Graduate of Lviv Polytechnic National University, khrystynaterletska1@gmail.com
ORCID: *0009-0002-7302-2625*

ENHANCING REAL-TIME DATA REPLICATION EFFICIENCY THROUGH OPTIMIZATION OF CHANGE DATA CAPTURE METHODS

Abstract. *Relevance of the research is driven by the growing need for efficient real-time data replication in distributed and hybrid environments, where system scalability, consistency, and low latency are critical. Traditional change propagation mechanisms are increasingly inadequate under high-frequency workloads, highlighting the need for a re-evaluation and modernization of Change Data Capture (CDC) methods.*

The aim *of the article is to provide a scientific rationale and develop approaches to improving the efficiency of real-time data replication through the optimization of Change Data Capture methods, taking into account the architectural and operational specifics of modern distributed computing environments.*

The research methodology *is based on systems analysis of CDC implementations in cloud and hybrid infrastructures, architectural modeling of replication flows, comparative evaluation techniques, typological classification of functional characteristics, and a criterion-based approach to assessing efficiency in various data update scenarios.*

The research results *include the identification of key functional features of CDC mechanisms, the classification of architectural replication models, and the substantiation of performance criteria such as latency, throughput, consistency, scalability, connectivity flexibility, and resource efficiency. Key technical constraints were identified in high-change-rate environments, particularly performance instability, access conflicts, and compatibility issues with traditional database systems and streaming platforms. It was demonstrated that event-driven architectures with asynchronous processing and adaptive buffering yield better performance when properly tuned.*

In the conclusions*, the relevance of hybrid CDC models is substantiated, the dependency between replication efficiency and architectural processing models is confirmed, and common implementation barriers in conventional database and stream-processing ecosystems are outlined.*

The research perspectives include the development of intelligent CDC strategies, dynamic change flow orchestration, and the improvement of cross-platform interoperability in multi-cloud infrastructures.

Key words: *asynchronous processing, streaming architectures, transaction consistency, change buffering, in-memory analytics.*

Христина ТЕРЛЕЦЬКА. ПІДВИЩЕННЯ ЕФЕКТИВНОСТІ РЕПЛІКАЦІЇ ДАНИХ У РЕЖИМІ РЕАЛЬНОГО ЧАСУ ЧЕРЕЗ ОПТИМІЗАЦІЮ МЕТОДІВ CHANGE DATA CAPTURE

Анотація. *Актуальність дослідження зумовлено необхідністю підвищення ефективності реплікації даних у режимі реального часу в умовах стрімкого зростання обсягів інформації, розширення розподілених інфраструктур і динамічного навантаження на системи обробки. Традиційні механізми передачі змін дедалі частіше виявляються недостатньо гнучкими або ресурсно затратними, особливо у висококонавантажених середовищах, що актуалізує завдання переосмислення методів Change Data Capture (CDC).*

Метою статті *є наукове обґрунтування та розробка підходів до підвищення ефективності реплікації даних у режимі реального часу шляхом оптимізації методів Change Data Capture з урахуванням специфіки сучасних розподілених обчислювальних середовищ.*

Методологія *дослідження базується на системному аналізі CDC-процедур у хмарних та гібридних інфраструктурах, моделюванні архітектурних схем реплікації, застосуванні методів порівняльного оцінювання, типологізації функціональних характеристик, а також критеріального підходу до визначення ефективності процедур у різних сценаріях оновлення даних.*

Результати *дослідження відображаються у визначенні ключових функціональних ознак CDC-реалізацій, класифікації архітектурних моделей реплікації, обґрунтуванні релевантних критеріїв оцінки (затримка, узгодженість, масштабованість, пропускна здатність, гнучкість підключення), а також у виявленні основних технологічних обмежень у висококонавантажених середовищах. Встановлено, що подієво-орієнтовані архітектури з асинхронною обробкою змін демонструють кращу продуктивність за умов правильної буферизації і контролю черг подій.*

У висновках *обґрунтовано доцільність гібридного підходу до CDC у складних архітектурах, підтверджено залежність ефективності від поєднання архітектурної моделі та режиму обробки, а також окреслено типові технічні обмеження, що виникають у процесі впровадження CDC у традиційні СКБД і потокові сервіси.*

Перспективами подальших досліджень визначено розробку інтелектуальних CDC-стратегій, адаптивного керування потоками змін і розширення міжплатформної сумісності компонентів у мультихмарних середовищах.

Ключові слова: *асинхронна обробка, потокові архітектури, узгодженість транзакцій, буферизація змін, аналітика в оперативній пам'яті.*

Problem statement. In today's environment of rapid growth in data volumes and demands for fast information processing, ensuring effective real-time data replication is becoming particularly important. Multi-organizational infrastructures, distributed computing systems, and cloud services require constant synchronous exchange of updates between sources and analytical platforms, which necessitates high-performance methods for detecting changes in data. Change Data Capture (CDC) technology is a key tool for detecting, capturing, and transferring changed data without the need to scan the entire source. However, in practice, its effectiveness is limited by a number of technical and logistical factors, including unstable performance when processing large transaction volumes, lack of universal compatibility with different types of data sources, difficulties in reconciling changes in distributed environments, and transmission channel overload. These challenges are particularly acute in systems where the delay between the occurrence of a change and its reflection in the target environment is critical, such as in financial, medical, or logistics platforms. In view of this, the scientific task is to find models that can adaptively respond to load changes, minimize duplicate operations, optimize resource consumption, and ensure transaction consistency. The practical significance of the research lies in the creation of CDC mechanisms that can be integrated into existing infrastructures without loss of compatibility, while complying with requirements for scalability, continuity, and transactional integrity, which has a direct impact on the stability and efficiency of real-time information systems.

Analysis of recent studies and publications. Analysis of current research on improving the efficiency of real-time data replication through optimization of Change Data Capture (CDC) methods allows us to identify four main areas of focus.

The first area focuses on the development of theoretical and applied concepts of CDC as a key replication tool. In their work, L. Hao, T. Jiang, Y. Lin, and Y. Lu [8] propose a classification of approaches to solving the CDC problem from the perspective of change source types–transaction logs, triggers, and timestamps. D. Seenivasan and M. Vaithianathan [14] analyze in detail the adaptation of CDC to the architecture of modern real-time computer systems, emphasizing the balance between the frequency of change capture and the load on the system. F. M. Imani, Y. D. L. Widyasari, and S. P. Arifin [12] consider CDC as a means of optimizing ETL processes, in particular to reduce data duplication and speed up analytical processing. T. Zheng, G. Chen, X. Wang, C. Chen, X. Wang, and S. Luo [20] explore CDC in the context of intelligent real-time big data processing, demonstrating the effectiveness of a coordinated environment for interaction between data sources, processing platforms, and applications. H. Chandra [4] conducted an experimental comparison of the effectiveness of CDC implementation in different types of data structures, allowing CDC to be adapted to the specifics of storage systems. Further research into adaptive algorithms for constructing change dependency graphs to optimize the CDC process in large distributed systems is promising.

The second area focuses on the security aspects of data replication and ensuring protection when implementing CDC on the front end. In his publication, Y. Horbenko [11] develops the concept of a secure automated framework with built-in encryption on the client side and implementation of the Zero Trust approach to API, which minimizes the risks of confidentiality loss during replication. In a subsequent study, Y. Horbenko [10] shows how the use of confidential computing technologies, in particular secure computing enclaves and homomorphic encryption, increases the security of front-end components during the exchange process. A promising direction in this area is the development of CDC processes with end-to-end encryption and anonymization at the source level, which complicates unauthorized analysis of change streams.

The third area covers architectural and infrastructure solutions for implementing CDC in cloud and distributed environments. R. B. Yurinec and I. B. Pirko [1] proposed innovative methods for integrating CDC into data warehouse filling procedures, emphasizing adaptability to changes in sources. D. Leschert, K. Sommerstad, J. Janzen, and C. Wester [3] analyze the difficulties of configuring CDC in industrial distributed environments, where configuration accuracy critically affects replication quality. B. Vagadia [17] considers CDC in the broader context of digital disintermediation and business process transformation, where replication performs the function of real-time updating of digital copies. In W. Q. Yan's monograph [18], CDC is described as a basic element of streaming data collection in intelligent video surveillance systems, which requires high performance and fault tolerance. A. Zakiah, R. Yusuf, and A. S. Prihatmanto [19] analyze the role of CDC in ensuring high data availability in the digital age at the conference. The development of CDC as a microservice with dynamic routing of changes in a multi-cloud environment with limited resources is promising.

The fourth area is related to the support of CDC in document management, analytics, and data governance within complex digital environments. The study by P. Dhakal, M. Munikar, and B. Dahal [6] presents an approach to automated data extraction from documents using a template-matching method, which can be integrated into CDC processes. The research conducted by P. A. Harris, R. Taylor, B. L. Minor, V. Elliott, M. Fernandez, L. O'Neal, L. McLeod, G. Delacqua, F. Delacqua, J. Kirby, and S. N. Duda [9] describes the work of the REDCap consortium, where CDC ensures the synchronization of changes in medical records across platforms. A promising direction is the integration of CDC with semantic data tagging mechanisms for enhanced contextual analysis in healthcare and public administration systems.

Despite significant progress in the field of incremental data loading, a number of aspects of the overall problem remain unresolved. In particular, the specifics of working with streaming sources in a cloud environment, where the dynamics of changes, uneven data flow, and parallel access conflicts create significant difficulties for stable processing, have not been sufficiently researched. Existing approaches do not fully address the need for transactionality, consistency, and version control in distributed analytics systems. There is also a lack of uniform criteria for evaluating the effectiveness of incremental update procedures in terms of performance, scalability, and data integrity. Much of the research relies on limited experimental scenarios or demonstrates applied implementations without proper generalized analytics.

The proposed study aims to partially fill these gaps by systematically analyzing theoretical and applied approaches to incremental loading, identifying key limitations of their application in cloud environments, and substantiating criteria for the effectiveness of adaptive change processing. Particular attention is paid to the potential of using platforms such as Delta Lake from the perspective of transactional updates, version control, and integration with streaming mechanisms. This allows us to expand the existing understanding of the functional capabilities of CDC in complex infrastructures and form a basis for further applied research and project solutions.

The **purpose of this article** is to provide scientific justification and develop approaches to improving the efficiency of real-time data replication by optimizing Change Data Capture methods, taking into account the specifics of modern distributed computing environments.

To achieve this goal, the following tasks are to be performed:

1. Analyze technical approaches and architectural models for implementing CDC in cloud and hybrid environments in order to assess their impact on latency and computing costs.

2. Justify the criteria for the effectiveness of CDC procedures, taking into account source types, processing modes, consistency, and scalability.

3. Identify problems with the application of CDC in high-load systems and formulate recommendations for their elimination based on adaptive buffering, asynchronous logging, and in-memory processing.

Presentation of the main material. CDC is the basis of modern synchronization processes in real-time systems. Its key purpose is to ensure fast and efficient transfer of only changed records from data sources to analytical platforms or target repositories without the need for a full table scan. This approach reduces latency, optimizes the use of network and computing resources, and ensures data relevance in heterogeneous environments. The development of cloud computing, streaming analytics, and microservice architecture has led to the emergence of various technical implementations of CDC, which differ in data sources, integration mechanisms, and configuration flexibility (Tab. 1).

Table 1

Technical Approaches to CDC Implementation and Their Functional Characteristics

Technical Approach	Description of Operation Mechanism	Functional Features
Triggers in Database Management Systems (DBMSs)	Utilization of SQL triggers that respond to insert, update, or delete operations within tables	Rapid implementation in traditional DBMSs; ensures transactional consistency
Transaction Log Analysis	Captures changes directly from the DBMS transaction log (e.g., WAL in PostgreSQL)	Minimal impact on the data source; high accuracy of changes; preserves temporal sequence
Snapshot-Based CDC	Compares the current table state with a previous version using control values	DBMS-independent; suitable for legacy systems
Event Stream Processing	Detects changes through streaming platforms (e.g., Apache Kafka, Apache Pulsar, Debezium)	Supports scalable real-time change processing; integrates with analytical pipelines
Platform-Oriented CDC Services	Employs managed services (e.g., Google BigQuery Data Transfer, Snowflake Streams, Azure CDC)	Enables automation; integrates with cloud-based analytics services; reduces operational workload

Source: compiled by the author based on [3; 4; 5; 7; 13]

In real-world conditions, preference is usually given to combinations of approaches, in particular the combination of transaction log analysis with event processing using Apache Kafka or Debezium, which allows large volumes of data to be processed in real time without overloading the sources. This solution provides low latency, scalability, and asynchronous data delivery to multiple consumers [5]. In cloud environments, where the “data as a service” model prevails, platform solutions such as Google Cloud BigQuery or Snowflake are increasingly being used, in which CDC functions are implemented at the infrastructure level as managed services, which greatly simplifies the deployment of complex ETL (Extract, Transform, Load) processes and ensures resilience to peak loads [13]. In distributed architectures and microservice approaches, event-driven CDC is the most promising for building adaptive synchronization systems capable of responding to data changes with minimal processing costs [16].

In modern information systems, data replication using CDC technology has become an integral part of ensuring consistency, continuity of processing, and scalability of computing processes. This is especially true in cloud and hybrid environments, where data is located in different regions, different types of storage and services are used, and the load on systems changes dynamically. CDC minimizes the amount of data transferred by focusing only on changed records, which is especially important when processing large streams of information in real time. However, the effectiveness of such replication largely depends on the architectural model of its implementation, which must take into account not only the computing capabilities of the system, but also the requirements for latency, consistency, and infrastructure costs (Tab. 2).

Table 2

Architectural Models for Data Replication Using CDC in Cloud and Hybrid Environments

Architectural Model	Description of Structure and Component Interaction	Potential for Latency and Cost Optimization
Push-Based Replication Model	The data source independently initiates the transmission of changes to the replica or handler	Suitable for low-latency scenarios but requires a stable connection; limited flow control
Pull Model with Periodic Polling	A CDC agent or replica periodically queries the source for updates	Lower processing overhead but higher latency, especially with large data volumes
Event-Driven Model with Event Broker	Changes are written to a log (e.g., Kafka), from which they are delivered to consumers	Highly scalable; ensures low latency; enables real-time event processing
CDC via Embedded Cloud Services	Replication is managed through cloud-based platforms (e.g., Google Dataflow, Azure Synapse, Snowpipe)	Minimal maintenance effort; optimized for cloud provider infrastructure

Source: compiled by the author based on [2; 6; 7; 17; 19; 20]

In practice, event-driven models are increasingly favored, wherein changes are captured in real time using event brokers such as Apache Kafka or Google Pub/Sub and then distributed to consumers or processors through streaming systems like Apache Flink or Google Dataflow [7]. This approach significantly reduces latency under high volumes of change while ensuring scalability and component isolation. Cloud-managed services, such as Snowpipe in Snowflake or CDC pipelines in Microsoft Azure Synapse, automate event processing by dynamically allocating computing resources based on actual workloads. This minimizes administrative overhead and helps prevent system downtime [15]. In complex hybrid architectures, hybrid models are being adopted more frequently. These allow for differentiated change capture frequencies based on data type or priority. For instance, critical transactions are recorded using event-driven methods, while secondary analytical data is updated in batches [2]. Consequently, the architecture of CDC-based replication must remain flexible and adaptive to meet contemporary demands for availability, processing speed, and infrastructure cost efficiency.

The effectiveness of implementing Change Data Capture (CDC) procedures in complex information environments depends not only on the choice of technology or tool, but primarily on the alignment of CDC mechanisms with data source types, processing modes, and specific update scenarios. There is no universal approach to CDC implementation, as different systems (transactional databases, analytical warehouses, streaming sources, or non-relational NoSQL systems) have distinct requirements regarding latency, consistency, and availability. In real-time contexts, it is particularly important to ensure consistency guarantees, adapt to variable workloads, and maintain scalability without compromising data integrity. Therefore, establishing clear criteria for evaluating the efficiency of CDC procedures is a necessary prerequisite for their optimal use in modern heterogeneous architectures.

In practical settings, the implementation of CDC procedures often involves finding a compromise between latency and consistency, depending on the characteristics of the data source. For instance, financial transactional systems prioritize strict transactional consistency, where minor delays are acceptable, but any loss or inconsistency of changes is intolerable. In streaming architectures with high-frequency updates, the priority is minimizing latency, even at the cost of temporary incompleteness, which is later resolved by event processors [11]. In cloud environments, scalability and resource efficiency are of primary importance. Platforms such as Snowflake employ adaptive CDC mechanisms with integrated services like Snowpipe Streaming, which provide low latency and automatic load balancing under distributed conditions. This approach enables the integration of heterogeneous data sources and optimizes replication channels without requiring modifications to the architecture of the source system.

Table 3

Efficiency Criteria for CDC Implementation in the Context of Data Source Types, Processing Modes, and Update Scenarios

Efficiency Criterion	Description	Dependence on Source or Scenario
Transmission latency	The time between a change in the source and its appearance in the target system	Critical for streaming data and OLTP systems (Online Transaction Processing)
Throughput	The number of changes the system can process per unit of time	Important for replicating large data batches, archives, and batch updates
Change consistency	Guarantee of the correct order and completeness of changes in the target replica	Essential for financial and healthcare systems where incomplete replication is unacceptable
Scalability	The ability of the CDC architecture to support increasing data volume and number of consumers	Crucial for cloud environments and multi-component microservice architectures
Connectivity flexibility	The ability to integrate CDC with diverse sources including relational, NoSQL, and API-based systems	Important for hybrid architectures, integration platforms, and ETL pipelines
Resource utilization efficiency	The ratio between processing costs and replication speed	A key parameter in systems with limited computing resources

Source: compiled by the author based on [1; 4; 6; 8; 10; 12; 20]

In systems with high-frequency data changes, the implementation of Change Data Capture (CDC) procedures encounters a range of systemic challenges that significantly affect the stability, reliability, and efficiency of real-time data processing. One of the primary issues is performance instability caused by the overloading of change transmission channels, especially during peak loads or when changes arrive unevenly. In high-transaction environments, CDC systems based on triggers or periodic scanning can place considerable strain on the source database, reducing overall throughput and increasing response times [16]. Even when using event-driven architectures with message brokers, there remains a risk of exceeding queue limits, which may result in delays or data loss under constrained computational resources [7].

Another critical issue involves conflicts arising from concurrent data access, which occur when multiple subsystems simultaneously read, process, and update records. In such cases, the integrity of the transactional context may be compromised, particularly in systems lacking centralized control over change streams. This becomes especially problematic in distributed databases or environments with microservice architectures, where CDC data consumers operate independently. Such independence can lead to state desynchronization or duplicated processing [8]. Even when transaction logs are used as the primary source of change data, it is not always possible to guarantee the correct order of events, especially in the presence of network delays or node failures [20]. An additional challenge lies in the compatibility limitations of CDC solutions with traditional database management systems (DBMS) and streaming platforms. Some legacy DBMSs, particularly those that do not support external access to transaction logs or rely on outdated locking mechanisms, significantly complicate the implementation of CDC without substantial infrastructure modification [5]. At the same time, many streaming platforms, such as Apache Flink or Amazon Kinesis, impose specific requirements on event formats and delivery order that do not always align with the capabilities of the data source. This creates the need for supplementary adapters, converters, or buffering layers, which increase system complexity, introduce potential points of failure, and reduce overall system stability [18].

Enhancing Change Data Capture (CDC) procedures in environments characterized by high-frequency changes and variable workloads requires the implementation of adaptive technological solutions capable of maintaining system resilience during peak loads, improving data consistency, and optimizing the use of computational resources. One of the key areas of improvement involves the adoption of adaptive change buffering mechanisms, which enable the dynamic adjustment of the volume of events accumulated prior to processing based on the current system state and available resources. Such buffering can be implemented either at the data source (for instance, in the form of commit logs with timestamps) or within the intermediate event broker layer, where intelligent queue management with prioritization is applied.

Another important component of CDC optimization is the use of asynchronous change logging. Offloading change capture to a separate process helps prevent blocking the main transaction processing flow, enables independent scaling of processing components, and reduces latency. Asynchronous change handling, when combined with timestamps and transaction identification mechanisms, ensures the orderly delivery of changes to replicated target systems or analytical services. This approach is particularly effective in systems with a high volume of parallel transactions, where synchronous CDC logging can negatively impact overall performance.

The third direction involves integrating CDC mechanisms with in-memory analytics, allowing modified data to be processed directly in main memory prior to final storage. This approach enables near-real-time analytics and forecasting without the need to wait for the completion of the full ETL cycle. In cloud environments, such integration is typically implemented through the use of in-memory computational clusters, such as Apache Ignite or Google BigQuery BI Engine, where modified data is delivered from CDC channels immediately following event processing. This makes it possible not only to reduce system response latency to data changes, but also to enable adaptive scaling of computational resources based on the context of the analytical query.

Collectively, these approaches provide a multi-level improvement in the efficiency of CDC procedures by reducing the impact of workload on performance, enhancing transactional consistency, minimizing analytical processing latency, and improving adaptability to the current demands of real-time distributed computing.

Conclusions. This study substantiates the role of Change Data Capture (CDC) technology as a key component of effective real-time data replication within modern cloud and hybrid architectures. It has been established that the efficiency of CDC implementation depends not only on the technical solution itself but also on its alignment with the type of data source, the nature of data changes, and the requirements for consistency, scalability, and latency. The analysis identified the most relevant architectural models for CDC deployment, described the criteria for their effectiveness, and examined technical constraints that arise in systems with high-frequency data changes. The main challenges include performance instability, conflicts during concurrent data access, and compatibility issues with certain database management systems and streaming platforms. The study provides a scientific rationale for the adoption of adaptive buffering mechanisms, asynchronous change logging, and integration with in-memory analytics as strategic directions for enhancing the reliability and performance of CDC infrastructures. Future research should focus on intelligent strategies for the dynamic optimization of CDC processes based on workload context and business process characteristics.

Bibliography:

1. Юринець Р. Б., Пірко І. Б. Інноваційні методики інтегрування даних для оптимізації процесу наповнення сховища даних. *Науковий вісник НЛТУ України*. 2024. Вип. 34, № 6. С. 101–105. DOI: https://doi.org/10.36930/40340614.
2. Beyond Teradata Migration: Implement a Modern Data Warehouse in Azure Synapse Analytics. *Microsoft Azure: website*. 2023. URL: https://learn.microsoft.com/en-us/azure/synapse-analytics/migration-guides/teradata/7-beyond-data-warehouse-migration (date of access: 13.06.2025).
3. Capture Configuration Challenges. 2023 IEEE IAS Petroleum and Chemical Industry Technical Conference (PCIC), New Orleans, USA. IEEE, 2023. P. 195–203. DOI: 10.1109/PCIC43643.2023.10414316.
4. Chandra H. Experimental results on change data capture methods implementation in different data structures to support real-time data warehouse. *International Journal of Business Information Systems*. 2020. Vol. 34, No. 3. P. 373. DOI: https://doi.org/10.1504/ijbis.2020.108651 (date of access: 13.06.2025).
5. Debezium Documentation. *Debezium: website*. 2024. URL: https://debezium.io/documentation (дата звернення: 06.06.2025).
6. Dhakal P., Munikar M., Dahal B. One-Shot Template Matching for Automatic Document Data Capture. 2019 Artificial Intelligence for Transforming Business and Society (AITB), Kathmandu, Nepal. IEEE, 2019. P. 1–6. DOI: 10.1109/AITB48515.2019.8947440.
7. Event-Driven Architecture. Confluent Developer: website. 2024. URL: https://developer.confluent.io/courses/microservices/event-driven-architecture/ (date of access: 13.06.2025).
8. Hao L., Jiang T., Lin Y., Lu Y. Methods for Solving the Change Data Capture Problem. In: Xiong N., Li M., Li K., Xiao Z., Liao L., Wang L. (eds). *Advances in Natural Computation, Fuzzy Systems and Knowledge Discovery*. ICNC-FSKD 2022. *Lecture Notes on Data Engineering and Communications Technologies*. Cham: Springer, 2023. Vol. 153. P. 1025–1034. DOI: https://doi.org/10.1007/978-3-031-20738-9_87.
9. Harris P. A., Taylor R., Minor B. L., Elliott V., Fernandez M., O'Neal L., McLeod L., Delacqua G., Delacqua F., Kirby J., Duda S. N. The REDCap Consortium: Building an International Community of Software Platform Partners. *Journal of Biomedical Informatics*. 2019. Vol. 95. Article 103208. DOI: https://doi.org/10.1016/j.jbi.2019.103208.
10. Horbenko Y. Confidential Computing in Front-End: Enhancing Data Security with Secure Enclaves and Homomorphic Encryption. *International Journal of Advanced Multidisciplinary Research and Studies*. 2025. Vol. 5, No. 3. P. 308–321. URL: https://www.multiresearchjournal.com/admin/uploads/archives/archive-1747130538.pdf (date of access: 13.06.2025).
11. Horbenko Y. Secure Front-End Automation Framework: A Novel Approach to Client-Side Data Encryption and Zero Trust API Interaction. *Asian Journal of Research in Computer Science*. 2025. Vol. 18, No. 6. P. 177–193. DOI: https://doi.org/10.9734/ajrcos/2025/v18i6690.
12. Imani F. M., Widyasari Y. D. L., Arifin S. P. Optimizing Extract, Transform, and Load Process Using Change Data Capture. *6th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)*, Batam, Indonesia. IEEE, 2023. P. 266–269. DOI: 10.1109/ISRITI60336.2023.10468009.

13. Import Data into a Secured BigQuery Data Warehouse. *Google: website*. 2023. URL: https://cloud.google.com/architecture/blueprints/confidential-data-warehouse-blueprint (date of access: 13.06.2025).

14. Seenivasan D., Vaithianathan M. Real-Time Adaptation: Change Data Capture in Modern Computer Architecture. *ESP International Journal of Advancements in Computational Technology*. 2023. Vol. 1, No. 2. P. 49–61. URL: https://www.researchgate.net/publication/381225264 (date of access: 13.06.2025).

15. Snowpipe Streaming – Snowflake Documentation. *Snowflake: website*. 2024. URL: https://docs.snowflake.com/en/user-guide/data-load-snowpipe-streaming-overview

16. Track Data Changes (CDC) in SQL Server. *Microsoft: website*. 2024. URL: https://learn.microsoft.com/en-us/sql/relational-databases/track-changes/track-data-changes-sql-server (дата звернення: 06.06.2025).

17. Vagadia B. Data Capture and Distribution. In: *Digital Disruption. Future of Business and Finance*. Cham: Springer, 2020. P. 45–66. DOI: https://doi.org/10.1007/978-3-030-54494-2_4.

18. Yan W. Q. Introduction to Intelligent Surveillance: Surveillance Data Capture, Transmission, and Analytics. Cham: Springer, 2019. 425 p. URL: https://books.google.com.ua/books?id=pheJDwAAQBAJ (date of access: 13.06.2025).

19. Zakiah A., Yusuf R., Prihatmanto A. S. The Benefits of Change Data Capture in Enhancing Data Availability in the Digital Transformation Era. *Widyatama International Conference on Engineering 2024 (WICOENG 2024)*. Atlantis Press, 2024. P. 302–308. DOI: https://doi.org/10.2991/978-94-6463-618-5_32.

20. Zheng T., Chen G., Wang X., Chen C., Wang X., Luo S. Real-time intelligent big data processing: technology, platform, and applications. *Science China Information Sciences*. 2019. Vol. 62. Article 82101. DOI: https://doi.org/10.1007/s11432-018-9834-8.

Дата надходження статті: 17.06.2025
Дата прийняття статті: 23.06.2025
Опубліковано: 23.09.2025