International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Challenges in Streaming ETL Pipelines for High-Frequency Data Ingestion and Real-Time Processing

Shiva Kumar Vuppala[1], Manohar Reddy Sokkula[2]

[1]IEEE Member | McNeese State University, Texas, USA | kumarvuppala.shiva@gmail.com ,
[2]IEEE Member | Northwest Missouri State University, Georgia, USA | Manohar.sokkula@gmail.com

ABSTRACT
The transition from traditional ETL (Extract, Transform, Load) pipelines to streaming ETL pipelines seeks to enhance the accessibility and usability of data by everyone in the organization, including non-technical organizational members. Traditional batch ETL pipelines, which process data at the end of a batch cycle, are ETL unsuitable for high-frequency data ingestion and real-time applications. Streaming ETL handles data in real-time as it is generated and accessed, allowing continuous processing of high-frequency data as it arrives. The paper analyzes the challenges in designing and implementing streaming ETL pipelines for high-frequency data ingestion and real-time processing. A review of existing literature led to the identification of the major streaming ETL challenges and their solutions. Some of the major challenges in streaming ETL pipelines include the ingestion of high-frequency data, achieving low-latency data transformation, recovering from faults, privacy and security, real-time data visualization, and availability of the required skills. Some strategies for addressing the challenges include backpressure and buffering, distributed messaging systems, windowing, lightweight serialization formats, checkpointing, idempotent operations, and watermarking. Organizations should address these challenges to unlock the full potential of streaming ETL pipelines.

INDEX TERMS Streaming ETL, real-time data processing, latency optimization, out-of-order data, backpressure management, checkpointing, throughput.

I. INTRODUCTION

A. CONTEXT

The rise of the value of data in organizations into major assets has characterized the last two decades. Some of the factors leading to this rise include the rise of digitization, which allows massive amounts of data to be generated and collected, Big Data analytics, the introduction of tools at the workplace that include large volumes of data, and the migration to cloud computing that enhances access to computing resources [1]. No one can deny the fact that collecting, processing, and analyzing data leads to informed decision-making and impacts the quality of customer relations through increased customer satisfaction. Organizations unlock the value of data by ensuring that the collected data remains relevant to business needs and complying with the FAIR principles, which include findable, accessible, interoperable, and reusable [2]. This has caused organizations to invest in the establishment of internal catalogs that provide a view of datasets with the support of automated pipelines that support the extraction of data from

multiple data sources. The goal is to enhance the accessibility and usability of data by everyone in the organization, including non-technical organizational members.

Organizations seek to meet these demands by implementing ETL (Extract, Transform, Load) pipelines. Traditional ETL tools are designed to handle structured data while providing manual visual interfaces that make it easy to create transformation logic. However, their reliance on batch processing and restriction to a limited number of predetermined sources of data makes them less responsive to the data needs of contemporary organizations. The evolution of current data ecosystems renders traditional ETL tools inefficient due to the inability to address emerging challenges such as data diversity as well as new functionality requirements such as the conversion of datasets into monetizable data products [1]. To overcome these challenges, streaming ETL has emerged as a critical approach to contemporary data management. Unlike traditional ETL pipelines, streaming ETL pipelines promote continuous ingestion, transformation, and delivery of data while minimizing latency. It is regarded as a significant development in the ETL practice, especially in an era where milliseconds may significantly impact outcomes, such as detecting fraud and sentiment analysis in social media. Streaming ETL's major benefits include real-time insights, reduced latency scalability, improved data freshness, and continuous processing [3].

B. PROBLEM STATEMENT

While it is indisputable that streaming ETL offers a wide range of benefits, such as providing timely insights and actionable data. It is associated with various challenges and complexities not found in traditional batch ETL. The challenges of streaming ETL are exacerbated by high-frequency data environments due to factors such as scalability, fault tolerance, low latency, and high throughput [4]. It is a challenging task to ensure that streaming ETL pipelines operate effectively and efficiently in high-frequency environments.

C. OBJECTIVES AND PAPER SCOPE

The paper analyzes the challenges in designing and implementing streaming ETL pipelines for high-frequency data ingestion and real-time processing. While streaming ETL pipelines are acceptable to play a vital role in modern data-driven systems, their design and implementation are challenging and highly complex. The paper seeks to advance this field by analyzing the critical challenges involved, evaluating the limitations of current tools, proposing mitigation measures, and exploring future directions. The scope of the paper is limited to the ETL components for high-frequency and real-time environments. It does not delve into related topics such as edge computing and machine learning.

D. STRUCTURE OF THE PAPER

The remainder of the paper is organized as follows. Section II provides a high-level overview of streaming ETL pipelines, including their components and technological landscape. Section III provides a detailed discussion of the challenges involved in streaming ETL pipelines. Section IV addresses the mitigation strategies for the identified challenges in streaming ETL, while Section V focuses on open challenges and future research directions.

II. OVERVIEW OF STREAMING ETL PIPELINES

A. ETL SYSTEM

An ETL system refers to a system used by an organization to present the movement of data and transactional processes used to extract data from a wide range of data sources and communication protocols. It takes to take and expertise to design these processes in a manner that fosters efficiency in

data acquisition, transformation of data, and loading data into storage environments [4]. The ETL process plays a vital role in data management practices, including integrating and migrating data. It encompasses gaining access to data at different source locations, processing the data through cleaning, integrating, and staging, and delivering the transformed data to storage systems.

The traditional approach to ETL is the batch ETL system, which works well with big data, but its approach leads to increased latency between the generation of data and its availability for analysis. It involves processing data after a batch cycle, which is associated with high latency. Analysts are forced to work with humongous amounts of data, which makes batch ETL unsuitable for real-time applications. The large amounts of information managed in batch ETL systems exhaust the capacity of available resources, including computing capacity and storage capacity, which leads to inefficiencies and increased operational costs [3]. The limitations of batch ETL systems have led to the emergence of streaming ETL systems.

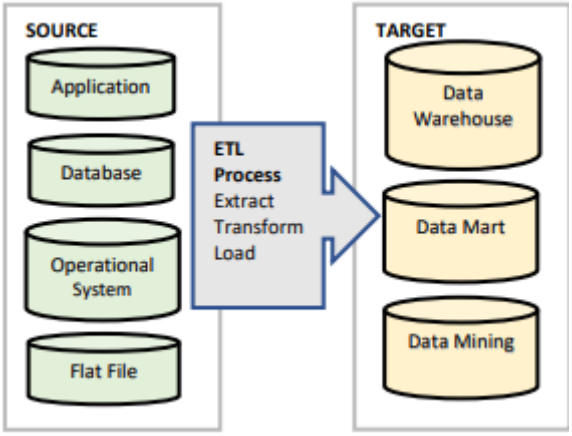

Figure 1: ETL System Flow in a Data Warehouse Environment

B. DEFINITION AND COMPONENTS OF STREAMING ETL

Streaming ETL is an evolution of the ETL practice in which data is continuously ingested through extracting, transforming, and loading processes for real-time data. Unlike batch ETH which handles data in large amounts, streaming ETL handles data in real-time as it is generated and accessed. It allows the continuous processing of data as it arrives, with an emphasis on minimizing or eliminating delays between the generation and analysis of data. Streaming ETL plays a crucial role in situations where real-time decision-making is essential, such as fraud detection and smart cities [5].

The components of streaming ETL include data sources, data ingestion, stream processing engine, real-time transformation, and data loading. Streaming ETL obtains data from multiple sources, including APIs, IoT devices, logs, sensors, and databases. These sources are capable of constantly generating information that requires real-time processing and analysis [3]. Data ingestion tools serve as a decoupling layer for absorbing information obtained from the above data sources for the establishment of its correctness and appropriateness for transfer to other systems. The stream processing engine fosters flexibility in stream operations, including transforming, updating, and consolidating data within the system. The engine can be obtained from Apache or Apache Strom. The real-time transformation stage utilizes various data transformations to further process streaming data. One of the tasks involved is the elimination of unneeded data or consolidation of existing data to yield the desired results. The purpose of data transformation is to enhance the readiness of data for analysis and use in decision-making [3].

Finally, the data loading stage involves storing the transformed data in various data storage systems, such as data warehouses or data lakes, for further analysis, visualization, and reporting.

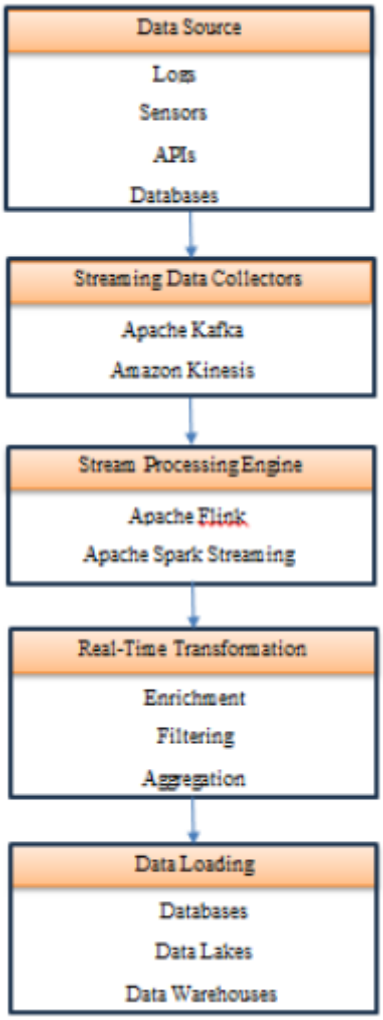

Figure 2: Streaming ETL Components

C. KEY FEATURES OF STREAMING ETL

1) HIGH AVAILABILITY

Streaming data are generated in an endless, constant flow, an indication that even the slightest disruption in their availability would significantly impact operations. The ability to deliver and replicate data help in ensuring the availability of data and preventing its loss [4]. Streaming ETL ensures that the data warehouse copes with streaming datasets and makes optimum use of the distribution and replication techniques to mitigate the risk of inconsistencies in the data.

2) LOW LATENCY

Data latency refers to the speed at which fresh data is delivered to meet business needs. Unlike in batch ETL systems, the time spent between the arrival of the data and their availability to users for real-time data is almost instant, which leads to low latency [6]. Streaming ETL is analyzed in real-time, which allows organizations to respond to changes as they occur quickly. Prioritizing low latency ensures that businesses have the latest analyses to allow timely decision-making.

3) HIGH SCALABILITY

The scalability of a data warehouse refers to its ability to adjust to sudden decreases or increases in the volume of data without compromising performance. Streaming ETL systems are highly scalable, which makes them capable of adjusting to changes in the volumes generated by their data sources [6]. They can scale horizontally to manage changes in loads and volumes without compromising performance. An online retail website can generate different data volumes over time, necessitating a highly scalable ETL system.

4) MINIMUM DISRUPTIONS

A streaming ETL system minimizes disruptions by exerting minimum burden on source systems during the extraction of data and utilizing efficient strategies for loading data to maintain the quality of service for data users in the organization. Streaming ETL systems process data as it is obtained to provide immediate updates and mitigate the delays that characterize batch jobs. The ingestion and transformation of data in real-time ensure that no downtime results from waiting for a batch before processing and loading new data [5]. Disruptions are minimized through enhanced system responsiveness, reduced resource spikes, fault tolerance, and smaller data chunks.

D. STREAMING ETL FRAMEWORKS AND TECHNOLOGIES

1) APACHE KAFKA

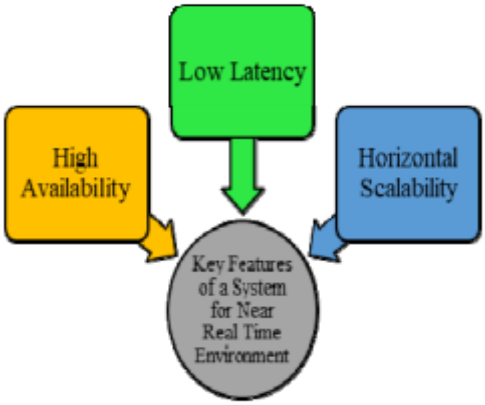

FIGURE 3: APACHE KAFKA ARCHITECTURE

Apache Kafka is a distributed streaming platform developed by Apache Software Foundation. It is designed for data pipelines that offer high throughput and low latency. It effectively handles humongous amounts of datasets, making it one of the most commonly used real-time data processing systems [7]. It stores and processes fault-free information streams. It works with the help of the producer API, consumer API, streams API, and connector API. It continuously accepts requests for sending and receiving messages.

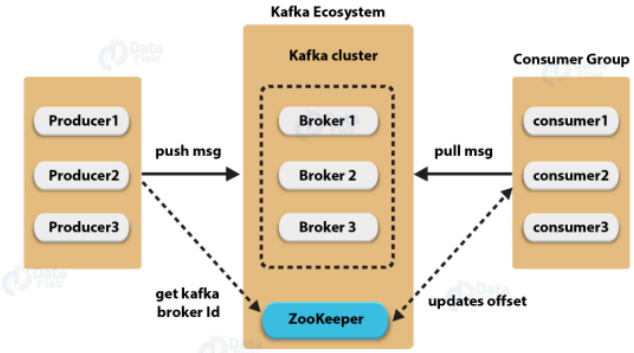

Figure 4: Apache Kafka Architecture

2) APACHE FLINK

Apache Flink is a streaming ETL processing framework capable of supporting event-driven applications while offering the capability of a single application to prevent the duplication of data [5]. Its pipelines can support stateful computations and effectively handle stream data.

3) APACHE STORM

Just like Apache Flink, Apache Storm is a free, open-source, distributed, real-time computational system for processing unbounded data streams. It is a stream processing technology that easily integrates with existing database and queueing technologies [8]. Its primary features include scalability and fault tolerance.

4) GOOGLE DATAFLOW

Google Dataflow is a data streaming service provided by Google Cloud Platform (GCP). The framework based on the Apache Bean programming model, which supports both batch and streaming data processing pipelines. It automatically handles resource provisioning, scaling, and management [9]. It integrates with Google Big Query and other GCP services to offer capabilities for handling data in real-time. It is suitable in situations requiring agile and scalable solutions for streaming ETL within the Google Cloud ecosystem.

5) OTHER TECHNOLOGIES

Two of the major tools used in streaming ETL include dbt (Data Build Tool) and Confluent KSQL. These two tools play a vital role in the transformation of raw data into meaningful, actionable formats that an organization can easily use [10]. Data Build tool is increasingly used in real-time pipelines that require integration with streaming platforms. It is SQL-based transformations and supports both collaborative and version control workflows. Confluent KSQL is also an SQL-based tool for real-time transformations that allows the definition of ETL workflows without requiring complex code. It is suitable for Kafka-centric architecture.

III. CHALLENGES IN STREAMING ETL PIPELINES

A. HIGH-FREQUENCY DATA INGESTION

Unstructured data contain different formats of high-volume data obtained from a wide range of data sources such as e-commerce platforms, online communities, electronic mail (e-mail), and social media. Such data includes a combination of different forms such as text, audio, video, and pictures. These features make high-frequency data ingestion a difficult task. First, high-frequency data ingestion systems must ensure data availability at the right locations and systems with minimal delay [11]. Any

latency could negatively impact real-time insights. These ingestion systems are required to handle sudden fluctuations in data volume and velocity without compromising performance, which can be a challenging task. The design and implementation of architectures capable of dynamically scaling resources can be a complex process. Implementing robust error detection, rollback, and recovery mechanisms in high-frequency data ingestion systems can be a major challenge.

B. LOW-LATENCY DATA TRANSFORMATION

Achieving low-latency data transformation in streaming ETL pipelines can be a critical challenge, especially because the system must process data in real-time and handle complex transformations at scale [3]. Data transformation in streaming ETL pipelines often involves complex transformations such as implementing machine learning models, which would require significant computational resources. It is challenging to meet the requirements for complex transformations without adding latency. Additionally, maintaining a state is one of the most important requirements in many streaming transformations. It is not an easy task to efficiently manage a state during high throughput conditions while avoiding delays. Finally, the coordination of transformations across distributed systems due to the increase in the velocity and volume of data may result in increased synchronization, which could compromise the need to maintain latency as low as possible.

C. FAULT TOLERANCE

The functioning of a streaming ETL pipeline requires that the system remains available for real-time data processing, which can be adversely affected by system failure. This could be failures occurring in the software or hardware, which could lead to loss of data or render the system inaccessible [3]. Any failures occurring could result in the loss of significant amounts of data at any of the stages of the ETL process. Additionally, failures occurring during data transformation and processing could lead to inconsistencies or the delivery of incorrect data. The transformation and loading processes could be negatively impacted by the arrival of out-of-order data, which could result from failures at any of the process stages. Finally, failures occurring in external systems could disrupt the streaming ETL process.

D. PRIVACY AND SECURITY

Streaming ETL pipelines may compromise the requirements for maintaining the confidentiality, integrity, and availability of data. The real-time extraction, transformation, and loading of data may compromise the privacy and security protections that should be accorded to individual health information, bank account numbers, online shopping membership, and identifiable social media information [11]. The risk of privacy violation is exacerbated by the practice of irregularly accumulating personal data across multiple social media platforms. While access to private information could aid criminal investigations, it is illegal and unethical to distribute or use other people's personal information without their consent. Real-time data processing makes it difficult to evaluate data for security and privacy concerns before it is transformed and loaded into storage systems.

E. DATA VISUALIZATION

Data visualization plays a vital role in enhancing the understandability of data. It helps in ensuring that information is reported clearly and effectively to its users through the use of a variety of data visualization techniques, such as animations, charts, tables, and graphs [11]. Unlike in traditional batch processing where it is easier to visualize stored or aggregated data, streaming ETL necessitates continuous, real-time analysis and visualization. It can be overwhelming to keep updating visualizations when the system used is incapable of handling high-frequency updates. Network delays and variability in processing times could lead to out-of-order data, which lead to inaccuracies and inconsistencies when

visualizations are developed in real-time.

F. SKILL REQUIREMENTS

Skill requirements are a major challenge in the design and implementation of streaming ETL pipelines, which are complex and require a multidisciplinary approach. Organizations experience difficulties in recruiting, training, and retaining professionals with the necessary skills, knowledge, and expertise for designing, implementing, and managing streaming data environment [11]s. The mastery of streaming ETL pipelines requires advanced knowledge and skills in real-time data processing frameworks such as Apache Flink and Apache Kafka [12]. An effective streaming ETL pipeline professional also requires programming and scripting skills, data modeling and transformation skills, data engineering knowledge, and the ability to integrate streaming ETL pipelines with external systems.

IV. MITIGATION STRATEGIES FOR STREAMING ETL CHALLENGES

A. STRATEGIES FOR HIGH-THROUGHPUT INGESTION

The challenge of high-throughput ingestion in streaming ETL can be mitigated through backpressure and buffering. Backpressure ensures that the streaming ETL system does not bite off more than it can handle. It focuses on ensuring that new data is accepted only when the system can handle it adequately [13]. It seeks to enhance the reliability and robustness of the streaming ETL system to ensure alignment between production and consumption rates. It employs feedback mechanisms to regulate the rates and enable the system to remain stable, mitigate data loss, and allow it to handle sudden spikes in workloads. The incorporation of an adaptive buffering mechanism and graph compression technique helps mitigate the problem of high-throughput in streaming ETL systems [14]. It focuses on allocating buffer window on systems handling lots of simultaneous traffic.

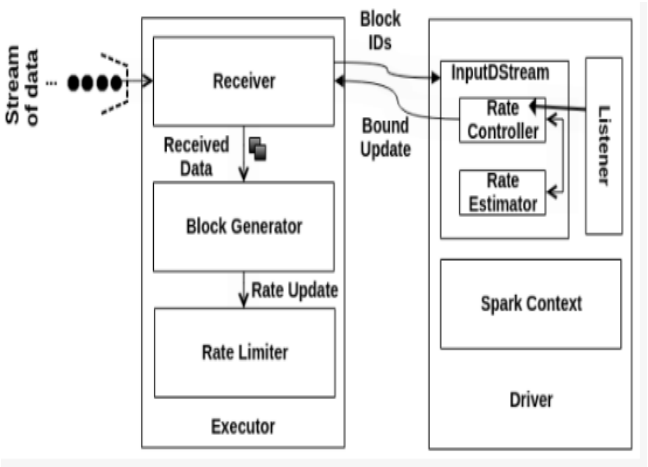

Figure 5: Apache Spark Backpressure Architecture

The use of distributed messaging systems could also help mitigate the challenge of high-throughput ingestion in streaming ETL. These systems enable high-throughput data ingestion by ensuring that message brokering remains scalable, fault-tolerant, and reliable. One of such systems is Apache Kafka, which operates at the streaming layer as a distributed Message Queuing System [15]. A data stream in Kafka is referred to as a topic, which is broken down and distributed to multiple nodes. The goal is to ensure that data ingestion tasks are distributed across multiple nodes or clusters, which leads to increased fault-tolerance, performance, and scalability during high-throughput data ingestion.

B. STRATEGIES FOR LOW-LATENCY PROCESSING

Windowing offers an effective solution to the problem of achieving low-latency processing in streaming ETL. It involves dividing the continuous stream of data into multiple windows or chunks to ensure that operations are applied over a subset of the data stream. Windowing is most effective in real-time data processing where timely insights are critical [16]. It ensures that the system achieves low-latency processing while offering continuous, real-time insights. Common windowing techniques include sliding windows, tumbling windows, and session windows. Utilizing shorter window durations mitigates delays occurring before the processing and consumption of data, but it may lead to increased computational overheads. The implementation of adaptive windowing ensures that window sizes are dynamically adjusted based on data patterns [16]. For instance, window sizes may be increased during periods of high activity.

Low-latency streaming may also be achieved by using lightweight serialization formats, which are optimized data formats for achieving efficient data transmission and storage. These formats are characterized by compactness, fast parsing and encoding, and support for schema evolution [17]. Common lightweight serialization formats include Apache Avro, Apache Parquet, Protobuf, and Message Pack. Using lightweight serialization forms leads to reduced data payloads, which is associated with increased transmission speed. The data processing speed is also accelerated due to the effect of optimized decoding and encoding on reducing CPU and memory usage.

C. FAULT TOLERANCE TECHNIQUES

One of the approaches to enhancing the tolerance of streaming ETL systems is checkpointing. The technique enables recovery from failures in streaming ETL systems while ensuring continuity and data consistency [18]. The state of a streaming ETL system is periodically saved to a durable storage system such that the application will not be required to restart from scratch but resume processing from the latest checkpoint after the occurrence of a failure. The system is equipped with the capability to periodically capture its internal state, write it to a durable storage medium, and read from the most recent checkpoint after the occurrence of a failure. Checkpointing can be synchronous when processing must be paused during the creation of a checkpoint, or asynchronous when the creation of a checkpoint does not require processing to be paused [18]. Some of the streaming ETL frameworks that incorporate checkpointing mechanisms include Kafka Streams and Apache Flink. Checkpointing promotes fault tolerance by enabling the streaming Etl system to recover quickly with minimal impact.

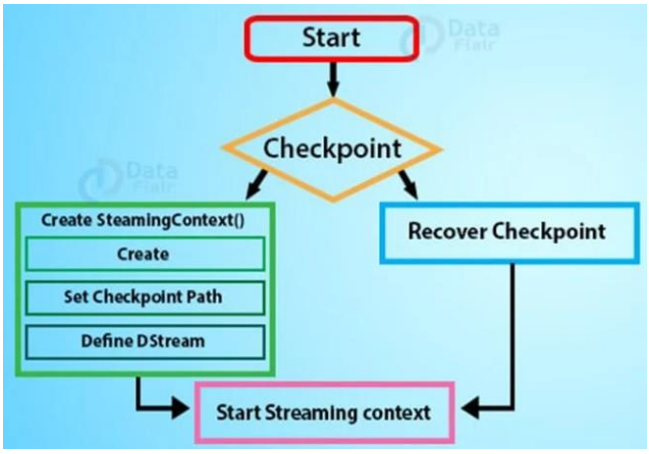

Figure 6: Spark Streaming Checkpoint in Apache Spark

Idempotent operations also play a vital role in enhancing fault tolerance in streaming ETL systems. Idempotence refers to the criterion that must be fulfilled by a streaming system in order to tolerate duplicate requests [19]. It seeks to ensure that different outcomes are not produced when the same event is processed multiple times to maintain consistency and data integrity while simplifying error handling. Three important idempotent operations include absolute value, closing a door, and HTTP DELETE request. Idempotence ensures that a retry after a failure of an operation does not produce inconsistencies or lead to duplication. The system will not require complex logic to find a way out when an operation is repeated, which leads to simplified failure recovery [18]. Data consistency is achieved through synchronization, repeatability, and consistency across distributed systems.

D. REAL-TIME DATA VISUALIZATION TECHNIQUES

One of the ways to manage the challenge of data visualization in streaming ETL pipelines involves employing mechanisms for handling out-of-order data. Network issues and distributed processing in streaming ETL pipelines may result in the arrival of out-of-order events, negatively impacting visualizations' accuracy [20]. The two important techniques for handling out-of-order events are windowing and watermarking. Windowing involves dividing the data stream into dynamic or fixed intervals based on either processing time or event time to allow the application of computation or aggregations function to every window. On the other hand, watermarking involves defining a boundary or threshold that indicates how much of the data arriving late can be accepted for every window [20]. For instance, the system may be configured to only accept data arriving within five minutes of the end of a window while discarding the rest. The watermarking feature of Apache Flink can effectively handle event-time data alignment issues to allow accurate data visualizations.

V. FUTURE RESEARCH DIRECTIONS

One of the areas for future research is the application of artificial intelligence (AI) in streaming ETL pipelines. The impact of AI on data pipelines is significant. Some of the areas that should be explored include the application of AI in automated error detection in streaming ETL systems, intelligent data transformation, and autonomous pipeline monitoring and management. Research in this area would highlight the potential of AI to revolutionize streaming ETL pipelines and unlock their full potential.

Another important research area is integrating streaming ETL with emerging technologies such as edge computing, quantum computing systems, and blockchain. Researchers should focus on how these integrations would enable the design and implementation of cutting-edge ETL systems for enhanced performance and capabilities.

VI. CONCLUSION

The implementation of streaming ETL pipelines is regarded as a novel approach to enhancing the accessibility and usability of data in organizations. They are viewed as a significant development in an era where milliseconds may significantly impact outcomes, such as detecting fraud and sentiment analysis in social media. The defining features of streaming ETL pipelines include high availability, low latency, high scalability, and minimum disruptions or fault tolerance. However, today's streaming ETL pipelines still face challenges regarding the ingestion of high-frequency data, achieving low-latency data transformation, recovering from faults, privacy and security, real-time data visualization, and availability of the required skills. Organizations should focus on addressing these challenges to unlock the full potential of streaming ETL pipelines. Some of the strategies for addressing the challenges

include backpressure and buffering, distributed messaging systems, windowing, lightweight serialization formats, checkpointing, idempotent operations, and watermarking.

REFERENCES

1. G. Zarate, M. J. L. Osa, A. I. Torre-Bastida, U. Iturraspe, J. Arjona, B. Navarro and A. Gimeno, "Evolution of Extract-Transform-Load (ETL) Processes Towards Data Product Pipelines," in 4th Eclipse Security, AI, Architecture and Modelling Conference on Data Space, New York, 2024.
2. M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg and G. Appleton, "The FAIR Guiding Principles for Scientific Data Management and Stewardship," Scientific Data, vol. 1, no. 1, pp. 1-9, 2016.
3. D. Seenivasan, "Real-Time Data Processing with Streaming ETL," International Journal of Science and Research (IJSR), vol. 12, no. 11, pp. 2185-2192, 2023.
4. A. Sabtu, N. F. M. Azmi, N. N. A. Sjarif and S. A. Ismail, "The Challenges of Extract, Transform and Load (ETL) for Data Integration in Near Real-Time Environment," Journal of Theoretical and Applied Information Technology, vol. 95, no. 22, pp. 6314-6322, 2017.
5. N. R. Mandala, "ETL in Big Data Architectures: Challenges and Solutions," Open Access | Double Blind Peer Reviewed Journal, vol. 13, no. 10, pp. 1061-1068, 2024.
6. A. Akambi and M. Masinde, "A Distributed Stream Processing Middleware Framework for Real-Time Analysis of Heterogeneous Data on Big Data Platform: Case of Environmental Monitoring," Sensors, vol. 20, no. 11, pp. 1-15, 2020.
7. R. S. Koppula, "Streamlining Data Ingestion with Apache Kafka and Databricks," Journal of Scientific and Engineering Research, vol. 10, no. 6, pp. 284-289, 2023.
8. P. Matai and A. Bhatia, "Stream Processing and Data Warehousing Integration," International Journal of Science and Research (IJSR), vol. 13, no. 9, pp. 1586-1590, 2024.
9. P. Borra, "Comparative Review: Top Cloud Service Providers ETL Tools - AWS vs. AZURE vs. GCP," International Journal of Computer Engineering and Technology (IJCET), vol. 15, no. 3, pp. 203-208, 2024.
10. R. Lakshmanasamy and G. Ganachari, "Integration of Dbt With Modern Data Stack Technologies," International Journal for Multidisciplinary Research (IJFMR), vol. 1, no. 1, pp. 1-6, 2023.
11. F. Gürcan and M. Berigel, "Real-Time Processing of Big Data Streams: Lifecycle, Tools, Tasks, and Challenges," in 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT), Ankara, Turkey, 2018.
12. S. Eeti, A. Jain and P. Goel, "Implementing Data Quality Checks In ETL Pipelines: Best Practices and Tools," International Journal of Current Science (IJCSPUB), vol. 10, no. 2, pp. 31-42, 2020.
13. H. Isah and F. Zulkernine, "A Scalable and Robust Framework for Data Stream Ingestion," in 2018 IEEE International Conference on Big Data (Big Data), Seattle, WA, USA, 2018.
14. K. Khan, "Enhancing Adaptive Video Streaming Through AI-Driven Predictive Analytics for Network Conditions: A Comprehensive Review," International Transactions on Electrical Engineering and Computer Science, vol. 3, no. 1, pp. 57-68, 2023.
15. E. Fernandes, A. C. Salgado and J. Bernardino, "Big Data Streaming Platforms to Support Real-time Analytics," in ICSOFT 2020 - 15th International Conference on Software Technologies, 2020.
16. T. Lyko, M. Broadbent, N. Race, M. Nilsson, P. Farrow and S. Appleby, "Improving Quality of Experience in Adaptive Low Latency Live Streaming," Multimedia Tools and Applications, vol. 83, no. 1, p. 15957–15983, 2024.

17. S. Jackson, N. Cummings and S. Khan, "Streaming Technologies and Serialization Protocols: Empirical Performance Analysis," IEEE Access, vol. 12, no. 1, pp. 25-39, 2024.
18. X. Wang, C. Zhanf, J. Fang, R. Zhang, W. Qian and A. Zhou, "A Comprehensive Study on Fault Tolerance in Stream Processing Systems," Frontiers of Computer Science, vol. 16, no. 2, pp. 1-20, 2022.
19. M. Fragkoulis, P. Carbone and V. Kalavri, "A Survey on the Evolution of Stream Processing Systems," The VLDB Journal, vol. 33, no. 1, p. 507–541, 2024.
20. T. Akidau, E. Begoli, S. Chernyak, F. Hueske, K. Knight, K. Knowles, D. Mils and D. Sotolongo, "Watermarks in Stream Processing Systems: Semantics and Comparative Analysis of Apache Flink and Google Cloud Dataflow," Proceedings of the VLDB Endowment, vol. 14, no. 2, pp. 3135 - 3147, 2021.