INTERNATIONAL JOURNAL OF SCIENTIFIC RESEARCH IN ENGINEERING AND MANAGEMENT (IJSREM)
VOLUME: 08 ISSUE: 05 | MAY - 2024 SJIF RATING: 8.448 ISSN: 2582-3930

Designing Efficient Data Pipelines: A Framework for Ingestion, Processing, and Enrichment

Mr.Kulkarni,Aryennh
Dept. Of Computer Science
RV *College of Engineering*
Benglauru,India
araajjheshk.cs20@rvce.edu.in

Dr. K Badrinath.
Dept. Of Computer Science
RV College of Engineering
Benglauru,India
badarinath.kb@rvce.edu.in

Abstract:ETL (Extract, Transform, Load) and ELT (E xtract, Load, Transform) pipelines are the foundation of m odern data processing, enabling efficient transformation a nd integration of various data sources. Despite extensive re search, significant gaps remain, particularly in the areas of data processing, data quality and integrity, and security a nd privacy. This article provides a comprehensive review o f the existing literature on prompts to identify gaps. We th en present the design and implementation of an advanced combination therapy system that addresses these issues. O ur pipeline combines real-time data streaming, efficient data processing, and advanc ed security features. The results of this study help create ef ficient, reliable, and secure ETL/ELT processes and provi de methods that organizations can use to optimize processe s and improve survey results.

Keywords—BigData, Pipeline, Extract-Transform-Load(ETL),

I. INTRODUCTION

The exponential increase in data volume and the complexities associated with making data processing efficient are pivotal challenges in the contemporary data landscape. ETL (Extract, Transform, Load) and ELT (Extract, Load, Transform) pipelines are fundamental components in this domain, facilitating the seamless transfer of data from diverse sources and its transformation into formats suitable for specific computational tasks. These pipelines play a crucial role in enabling organizations to handle large-scale data operations effectively.

However, despite their significance, numerous obstacles persist that impede the efficacy and performance of ETL and ELT processes. One major issue is ensuring data quality and integrity, which is essential for producing reliable analytical outcomes. Data inconsistencies and errors can severely impact the validity of analysis, making it imperative to develop robust mechanisms for data validation and correction. Furthermore, the heightened emphasis on data security and privacy, driven by stringent regulatory frameworks, adds another layer of complexity. Safeguarding data throughout the ETL/ELT process is paramount to maintain compliance and protect sensitive information.

In addition to these challenges, the demand for real-time data processing is escalating, particularly in applications that require instantaneous insights and rapid decision-making capabilities. Traditional ETL/ELT pipelines often struggle to meet these requirements due to their inherent latency and batch processing nature. This necessitates the exploration of innovative solutions that can accommodate real-time data ingestion and processing.

II. BACKGROUND

ETL (Extract, Transform, Load) and ELT (Extract, Load, Transform) pipelines are processes used in data integration to move data from its source to a destination such as a data warehouse or database. In ETL pipelines, data is first extracted from various sources, then transformed to fit the desired structure or schema, and finally loaded into the target system. This transformation step may involve cleaning, filtering, aggregating, or converting data types to ensure compatibility with the target system.

On the other hand, ELT pipelines follow a similar process but with a different sequence. Data is first extracted from sources and loaded directly into the target system without significant transformation. Transformation tasks are then performed within the target system, leveraging its processing power and capabilities. [

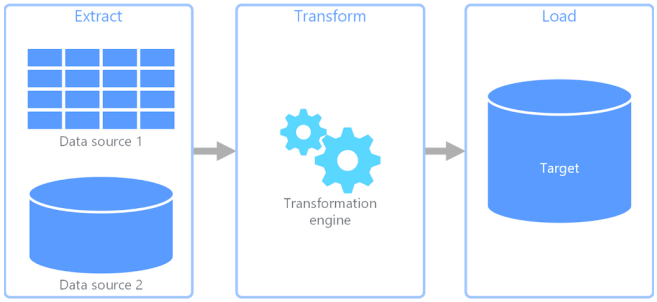

Fig0.1 ETL pipeline stages

[1] discusses provides a conceptual model of ETL pipeline to automate monitoring, fault detection. The model uses nodes and connectors to present the activities and their connections and data workflow. It emphasizes on designing fully automated fault tolerant network. Cloud providers offer a range of services and tools that enable the creation of data pipelines in a flexible and scalable manner. Leveraging services such as AWS Glue, Azure Data Factory, or Google Cloud Dataflow, users can easily extract, transform, and load data from various sources

into cloud-based storage or databases. These platforms provide pre-built connectors, data transformation capabilities, and scalable infrastructure, allowing organizations to efficiently orchestrate and automate complex data workflows while benefiting from the scalability and reliability of cloud computing resources. [2] explains the process of creating efficient data analysis pipelines. It postulates the creation of fully managed or proprietary cloud services for pipeline creation ex. Apache airflow. The size of a dataset significantly influences the architecture of a data pipeline. For small datasets, a simpler pipeline with fewer processing steps and lower resource requirements may suffice, often favoring straightforward ETL processes. In contrast, large datasets necessitate more sophisticated architectures capable of handling high volumes of data efficiently, typically involving distributed processing frameworks like Apache Spark or Hadoop. Scalability, fault tolerance, and parallel processing become crucial considerations to ensure optimal performance and reliability as dataset size increases. [3] explores the problem of creating pipelines for data science applications. It adopts a 3-way approach of analyzing 3 datasets of different sizes. The smaller pipelines tend to be more linear and lack several stages present in larger, complex pipelines. Data lakehouses merge the advantages of data lakes and data warehouses, offering a unified solution for storing and analyzing data. They combine the scalability and flexibility of data lakes, where data is stored in its original form, with the structured querying capabilities of data warehouses. This integration allows organizations to efficiently manage and analyze both structured and unstructured data, facilitating faster insights and decision-making processes. [5] explores the idea of data lakehouse architecture. It portrays the use of open direct-access data formats like Apache Parquet, for better performance and shows that optimizations like caching and data layout enhancements enable lakehouse systems to achieve high performances and exhibit that they are an improvement over traditional data warehouse architecture.[7] discusses the use of serverless platforms that allow customers to pay for resources based on use. It introduces TOSCA (Topology and Orchestration Specification for Cloud Applications) standard, and TOSCAdata which focuses on modelling data pipeline based cloud applications. *Data lakes serve as vast repositories for storing diverse types of data, ranging from structured to unstructured, in their native formats.* Unlike traditional databases, which require predefined schemas, data lakes allow organizations to ingest raw data at scale without immediate structuring. This flexibility enables data lakes to accommodate data from various sources, including IoT devices, social media platforms, and enterprise applications. By centralizing data in a single location, data lakes empower organizations to perform advanced analytics, machine learning, and other data-driven tasks, driving innovation and informed *decision-making.*

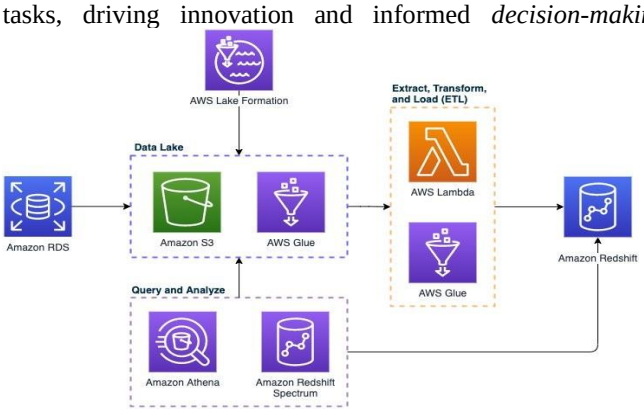

Fig0.2 Datalake architecture

[8] addresses the concept of data-lakes with a four zone approach, each with a defined and process layer. Metadata management plays a crucial role in data pipelines by providing essential context and insights about the data being processed. Metadata includes information about the structure, format, quality, and lineage of the data, enabling users to understand its meaning and relevance. In data pipelines, metadata management involves capturing, storing, and maintaining metadata throughout the data lifecycle. This includes metadata extraction from source systems, annotation during data transformation processes, and retention in data repositories. Effective metadata management ensures data lineage, enhances data quality, facilitates data discovery and governance, and supports compliance requirements. By centralizing and standardizing metadata across the pipeline, organizations can improve data visibility, interoperability, and decision-making capabilities, ultimately maximizing the value derived from their data assets. [9] explores the advantages and disadvantages of data lake systems and empathizes on the importance of management of metadata and argues for effective data governance and metadata querying issues. [10] showcases the use of Hadoop and its ecosystem to be the most used tools for constructing data lakes, indicating their prominence in current architecture. [11] gives an overview of ETL tools and compares Talend and Informatica, both of which are data integration tools. Data quality is paramount in data pipelines as it directly impacts the reliability and usefulness of the insights derived from the data. Ensuring high data quality involves several key aspects, including accuracy, completeness, consistency, and timeliness. Throughout the data pipeline, measures must be implemented to validate and cleanse incoming data, detect and correct errors, and maintain data integrity. This often involves implementing data validation rules, performing data profiling and cleansing techniques, and establishing data quality monitoring and governance processes. By prioritizing data quality throughout the pipeline, organizations can enhance the trustworthiness of their data-driven decisions, minimize the risk of errors and inaccuracies, and maximize the value derived from their data assets. [12] highlights the significance of data quality (DQ) in the ETL (Extract, Transform, Load) process crucial for Business Intelligence & Analytics (BI&A). It reveals limitations in existing tools that affect final data accuracy (Souibgui et al., 2019). Moreover, it underscores the increasing

complexity of big data and data-driven BI&A solutions, which exacerbates data quality challenges.

III. METHODOLOGY

The proposed pipeline in this paper is a three stage pipeline involving three processes namely Ingestion, Processing and Enrichment . The pipeline is built on Amazon Web Services (AWS) as a cloud provider to provision the required resources.

Technology used:

***Amazon S3*:** Amazon S3 (Simple Storage Service) buckets are storage containers provided by Amazon Web Services (AWS) for storing and managing data objects. These buckets are highly scalable, durable, and secure, offering a reliable solution for storing a wide range of data types, including documents, images, videos, and application data. S3 buckets can be accessed over the internet using a unique URL and offer features such as versioning, access control, and lifecycle management to help organizations manage their data efficiently. With S3 buckets, users can store massive amounts of data cost-effectively, while benefiting from high availability and durability provided by AWS infrastructure.

AWS Lambda: AWS Lambda is a serverless compute service offered by Amazon Web Services (AWS) that enables users to run code without provisioning or managing servers. With Lambda, developers can upload their code, specify the triggers that should execute it, and AWS takes care of the rest, automatically scaling and managing the infrastructure as needed. This event-driven architecture allows users to respond to events from various AWS services or custom triggers, such as HTTP requests or database updates, with minimal operational overhead. Lambda supports multiple programming languages, including Python, Node.js, and Java, making it accessible to a wide range of developers. By using Lambda, organizations can build highly scalable and cost-effective applications, paying only for the compute time consumed by their code.

AWS Eventbridge : AWS EventBridge is a fully managed event bus service provided by Amazon Web Services (AWS) that makes it easy to connect applications together using events. With EventBridge, users can create event-driven architectures by routing events from AWS services, SaaS applications, and custom sources to targets such as AWS Lambda functions, SNS topics, SQS queues, and more. EventBridge simplifies event management by offering features like schema discovery, event filtering, and event replay, allowing developers to focus on building scalable and resilient applications. By decoupling application components and integrating seamlessly with other AWS services, EventBridge enables organizations to create event-driven workflows that respond dynamically to changes in their environment.

AWS Glue: AWS Glue is a fully managed extract, transform, and load (ETL) service provided by Amazon Web Services (AWS) for preparing and transforming data for analytics and machine learning. With Glue, users can easily discover, catalog, and clean data from various sources, including databases, data lakes, and streaming platforms, without managing infrastructure. Glue automates much of the ETL process by generating code to extract, transform, and load data, reducing the time and effort required to prepare data for analysis. It also provides features like data deduplication, schema evolution, and job scheduling, allowing users to build scalable and reliable data pipelines. By using Glue, organizations can accelerate their data preparation workflows and enable faster and more accurate decision-making.

AWS DynamoDB: Amazon DynamoDB is a fully managed NoSQL database service provided by Amazon Web Services (AWS) that offers seamless scalability, high availability, and low latency for applications requiring single-digit millisecond response times. DynamoDB is designed to handle massive workloads with consistent, single-digit millisecond latency, making it suitable for a wide range of use cases, from web and mobile applications to gaming and IoT. With DynamoDB, users can store and retrieve any amount of data, and scale up or down automatically to accommodate fluctuating workloads. Its flexible data model supports both document and key-value data structures, and it offers features like encryption at rest, automated backups, and global tables for cross-region replication, ensuring data security and durability. By using DynamoDB, organizations can build highly scalable and responsive applications without worrying about managing the underlying infrastructure.

AWS Simple Notification Service: Amazon Simple Notification Service (SNS) is a fully managed messaging service provided by Amazon Web Services (AWS) that enables the sending and receiving of messages or notifications between distributed systems, microservices, and application components. With SNS, users can publish messages to topics, which act as communication channels, and subscribers can receive these messages through various protocols such as HTTP, HTTPS, email, SMS, and more. SNS simplifies the creation and management of pub/sub messaging architectures, allowing developers to decouple components, scale applications, and send notifications to multiple endpoints simultaneously. By using SNS, organizations can build highly resilient and scalable event-driven systems that react dynamically to changes in their environment, ensuring reliable message delivery and real-time communication across distributed systems.

PySpark: PySpark is a powerful open-source data processing framework that provides an interface for programming Spark with Python. Built on top of Apache Spark, PySpark enables users to process large-scale datasets in a distributed and parallel manner, leveraging the capabilities of Spark's in-memory computing engine. With PySpark, developers can write concise and expressive code to perform data manipulation, analysis, and machine learning tasks, using familiar Python syntax and libraries. PySpark seamlessly integrates with other Python data science tools and frameworks, such as pandas, NumPy, and scikit-learn, allowing for easy interoperability and extending its capabilities. By leveraging PySpark, organizations can efficiently process and analyze massive datasets, extract valuable insights, and build scalable data-driven applications, all within the Python ecosystem.

Terraform : Terraform is an open-source infrastructure as code (IaC) tool developed by HashiCorp that allows users to define and provision cloud infrastructure using declarative configuration files. With Terraform, users can describe their desired infrastructure in code using a simple and human-readable configuration language and terraform handles the provisioning and management of resources across multiple cloud providers, including AWS, Azure, and Google Cloud Platform. Terraform's state management and dependency graph enable safe and efficient infrastructure changes, allowing users to track and version infrastructure changes over time. By using Terraform, organizations can automate infrastructure deployment, maintain consistency across environments, and accelerate the development and delivery of cloud-native applications.

***Parquet** : Parquet is a columnar storage format optimized for big data processing. It organizes data by columns for efficient*

storage and query performance. With advanced compression and support for nested data structures, Parquet is widely used in analytical workloads across various big data frameworks.

Dataset : The proposed pipeline architecture uses an ecommerce dataset which contains 3 records namely – transaction data, customer data and product data. The dataset is generated automatically through python scripts which populate the dataset with randomly generated values. The pipeline largely deals with using the transaction data which is transmitted through the pipeline. The initial location for storing the transaction data in a AWS S3 bucket location. .The initial locations for the product and customer records are AWS DynamoDB tables.

Ingestion : Data ingestion serves as the foundational stage within a data pipeline, facilitating the collection and consolidation of diverse datasets into a centralized repository for subsequent processing and analysis. At its core, data ingestion entails extracting raw data from disparate sources, ranging from databases to files and streaming platforms, and orchestrating its seamless transfer into a designated storage or processing environment. Within the context of this pipeline, the data ingestion workflow operates as a recurring process, executed at predefined intervals, such as hourly increments. Leveraging the capabilities of Amazon S3 bucket service as the primary data source, data retrieval is orchestrated through the utilization of an AWS EventBridge resource. This resource governs the triggering of a Python script responsible for data extraction, a task seamlessly executed within the cloud environment by an AWS Lambda function. The source data, originally formatted in JavaScript Object Notation (JSON), undergoes conversion into the optimized Parquet format, enhancing storage efficiency and analytical performance. The resultant Parquet file is then stored within a separate S3 location, thereby preserving data integrity and accessibility. This entire data ingestion workflow is orchestrated and triggered at scheduled intervals by the EventBridge service, ensuring the timely and systematic ingestion of data into the pipeline.

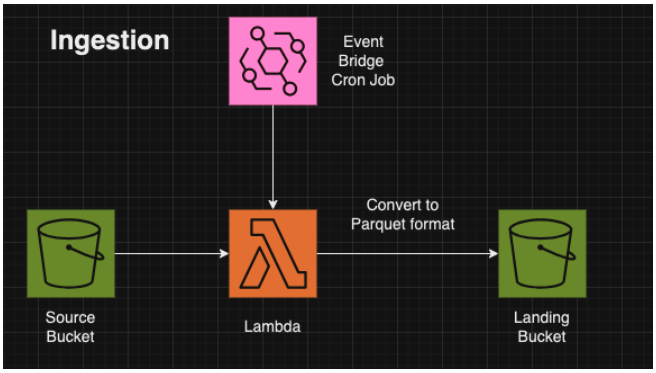

fig 0.3 Ingestion Process

Processing: Following successful data ingestion, the subsequent step within the pipeline involves processing the ingested data. Initially stored in Parquet format, the data is retrieved from the landing bucket, marking the commencement of the processing phase. An AWS EventBridge triggers in response to the creation of a new object in the landing bucket, serving as a signal that the ingestion process has concluded successfully. This EventBridge event, in turn, initiates an AWS Glue workflow designed to orchestrate further data processing tasks. The workflow encompasses a Glue trigger and a Glue job, wherein the latter undertakes the transformation of Parquet-formatted data into a Spark DataFrame. Subsequently, the Glue job executes a series of validation operations, including schema validation, data cleansing, deduplication, and data copying to another S3 location. Upon completion of these operations, the Glue job reverts the data back to Parquet format before storing it in the target S3 location. Additionally, the Glue job is tasked with creating Hudi tables for the cleaned dataset, enhancing data organization and accessibility. Finally, the Glue job facilitates the transfer of processed data to yet another S3 location, preparing it for subsequent stages within the pipeline. This cohesive processing workflow ensures the systematic validation, transformation, and storage of ingested data, laying the groundwork for advanced analytics and downstream processing tasks.

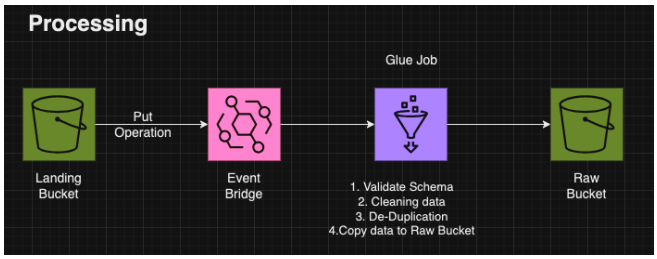

fig.0.4 Processing Stage

The enrichment phase follows the processing stage and is initiated upon the creation of an object in an S3 bucket, triggered by EventBridge. This event prompts the execution of a Glue job responsible for analyzing transaction data. Based on specific transactions, the Glue job updates customer profiles and adjusts product quantities in the products table accordingly. Leveraging data stored in DynamoDB tables, encompassing product and customer records, the Glue job orchestrates the generation of recommendations through a rule-based approach. It identifies products purchased by customers and their respective categories, subsequently compiling a list of related products from the product inventory. This curated dataset, mapping each customer to a list of recommended products, is then pushed to an S3 location, facilitating utilization in recommendation systems and marketing campaigns. This enriching process enhances customer engagement and personalization, driving targeted marketing efforts and optimizing product recommendations.

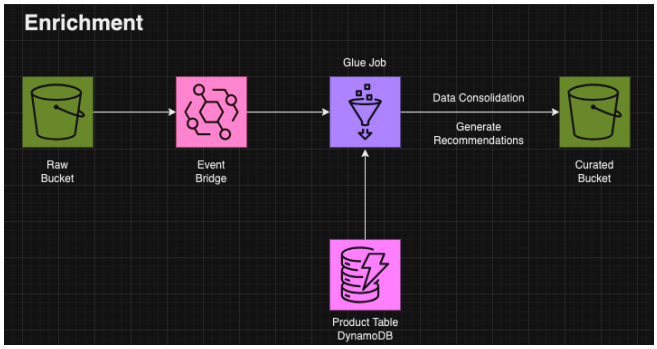

Fig.0.5 Enrichment Stage

Notification and Replication: Following the enrichment phase, the subsequent stage is triggered to process the generated recommendations. Initially, the recommendations derived from the previous phase are retrieved, and an AWS SNS topic is established. This notification service is instrumental in disseminating the list of recommendations and product analytics to pertinent stakeholders, including product owners, business intelligence teams, and marketing departments for targeted email campaigns. Additionally, as part of this stage, the replication of the original S3 bucket, which served as the data source, is conducted, with the replicated data stored in a separate bucket designated for marketing analytics. With these actions, the pipeline concludes, having successfully orchestrated the transformation of raw data into actionable insights, facilitating informed decision-making and strategic initiatives across the organization.

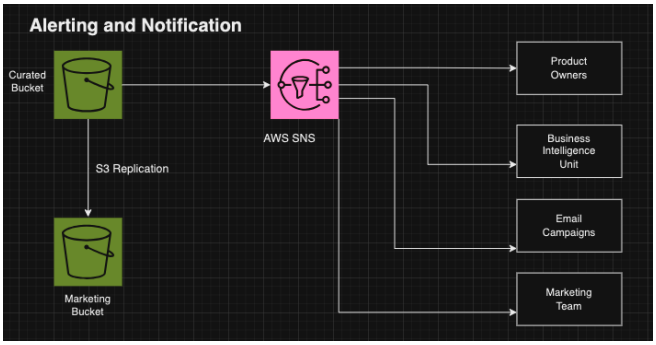

Fig.0.6 Notification and Replication

IV. RESULTS AND DISCUSSIONS

Ingestion : The ingestion process results in fetching data from the source s3 bucket through a python script running inside an AWS Lambda function, converting the data from a JSON to a parquet format, and then storing it in another s3 bucket location. This operation should occur every hour triggered by the Event bridge rule. The AWS metrics for each resource can be observed to check the number of invocations for a particular resource to confirm the frequency of its invocation. A similar pattern can be observed for the eventbridge metrics with the same frequency.

Processing : The processing phase results in a cleaned dataset being pushed into an S3 location. The processing phase is triggered by an Eventbridge whenever an object is created in the target location of the ingestion process. Therefore the frequency of triggering the processing phase and subsequent pipeline is same as the ingestion process. A successful processing phase can be inferred by checking the Event bridge rule invocation metrics to confirm if it is executed each hour .The Event bridge triggers a glue-workflow which contains the script to perform processing operation inside a gluejob. The successful running of the glue-workflow should have the same frequency as that of the Eventbridge triggering it.

Enrichment : Enrichment phase is triggered whenever the cleaned data is pushed into the target s3 location. This event is picked up by an eventbridge which triggers a gluejob. The successful invocation of eventbridge signals to to the enrichment process being created The glue workflow also is triggered at same frequency of the eventbridge

Notification and Replication: The notification is received through AWS Simple Notification Service(SNS). For testing purposes, the SNS topic is created for an email response which send an email every time data is received in the s3 bucket post enrichment.

V. CONCLUSIONS

The Ingestion, Processing and Enrichment pipeline is an efficient way to create fault tolerant pipelines. Individual units of the pipeline can be tested, which makes unit testing easier as each phase of the pipeline is independent of the previous phase and does not depend on the upcoming phase either. Processing data to convert into parquet format, data cleaning ensure that the amount of storage and therefore the costs associated with provisioning resources using cloud providers goes down. The pipeline ensures data redundancy is reduced by fetching data periodically and updating it in-store, reducing the required amount of storage space.

VI. REFERENCES

[1] Raj, A., Aiswarya, Bosch, J., Olsson, H., & Wang, T. (2020). Modelling data pipelines.

[2] Koivisto, T. (2019). Efficient data analysis pipeline.

[3] Biswas, S., Wardat, M., & Rajan, H. (2022). The art and practice of data science pipelines: A comprehensive study of data science pipelines in theory, in-the-small, and in-the-large. In Proceedings of the 44th International Conference on Software Engineering (ICSE '22) (pp. 2091–2103). Association for Computing Machinery. https://doi.org/10.1145/3510003.3510057

[4] Ranjan, R. (2014). Streaming big data processing in datacenter clouds. IEEE Cloud Computing, 1(1), 78-83. https://doi.org/10.1109/MCC.2014.22

[5] Armbrust, M., et al. (2021). Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics. Proceedings of CIDR, 8.

[6] Dabbèchi, H., Haddar, N. Z., Elghazel, H., & Haddar, K. (2021). Social media data integration: From data lake to NoSQL data warehouse. In A. Abraham, V. Piuri, N. Gandhi, P. Siarry, A. Kaklauskas, & A. Madureira (Eds.), Intelligent systems design and applications. ISDA 2020. Advances in intelligent systems and computing (Vol. 1351, pp. 747-755). Springer. https://doi.org/10.1007/978-3-030-71187-0_64

[7] Dehury, C. K., et al. (2022). TOSCAdata: Modeling data pipeline applications in TOSCA. Journal of Systems and Software, 186, 111164. https://doi.org/10.1016/j.jss.2021.11116

[8] Ravat, F., & Zhao, Y. (2019). Data lakes: Trends and perspectives. In Database and Expert Systems Applications (pp. 304–313). Springer. https://doi.org/10.1007/978-3-030-27615-7_23

[9] Sawadogo, P., & Darmont, J. (2021). On data lake architectures and metadata management. Journal of Intelligent Information Systems, 56(1), 97-120. https://doi.org/10.1007/s10844-020-00596-4

[10] Couto, J., Borges, O., Ruiz, D., Marczak, S., & Prikladnicki, R. (2019). A mapping study about data lakes: An improved definition and possible architectures. In Proceedings of the 31st International Conference on Software Engineering & Knowledge Engineering (SEKE 2019) (pp. 453-458). KSI Research Inc. and Knowledge Systems Institute Graduate School. https://doi.org/10.18293/SEKE2019-129

[11] Sreemathy, J., Brindha, R., Nagalakshmi, M. S., Suvekha, N., Ragul, N. K., & Praveennandha, M. (2021). Overview of ETL tools and Talend-data integration. In 2021 7th International Conference on Advanced Computing and Communication Systems (ICACCS) (pp. 1650-1654). IEEE. https://doi.org/10.1109/ICACCS51430.2021.9441984

[12] Souibguia, M., Atigui, F., Zammali, S., & Cherfi, S. (2019). Data quality in ETL process: A preliminary study. Procedia Computer Science, 159, 22-31. https://doi.org/10.1016/j.procs.2019.09.167

[13] Vyas, B. (2022, December 28). Optimizing data ingestion and streaming for AI workloads: A Kafka-centric approach.

[14] Giebler, C., Gröger, C., Hoos, E., Eichler, R., Schwarz, H., & Mitschang, B. (2021). The data lake architecture framework. In Proceedings of the BTW 2021 (pp. 351-370). Gesellschaft für Informatik. https://doi.org/10.18420/btw2021-19

[15] Hlupić, T., Oreščanin, D., Ružak, D., & Baranović, M. (2022). An overview of current data lake architecture models. In 2022 45th Jubilee International Convention on Information, Communication and Electronic Technology (MIPRO) (pp. 1082-1087). IEEE. https://doi.org/10.23919/MIPRO55190.2022.9803717

[16] Giebler, C., Gröger, C., Hoos, E., Schwarz, H., & Mitschang, B. (2019). Leveraging the data lake: Current state and challenges. In C. Ordonez, I. Y. Song, G. Anderst-Kotsis, A. Tjoa, & I. Khalil (Eds.), Big data analytics and knowledge discovery. DaWaK 2019. Lecture Notes in Computer Science (Vol. 11708). Springer. https://doi.org/10.1007/978-3-030-27520-4_13

[17] Singh, A., & Ahmad, S. (2019). Architecture of data lake. International Journal of Scientific Research in Computer Science, Engineering and Information Technology, 5(2), 4-4.

[18] Nambiar, A., & Mundra, D. (2022). An overview of data warehouse and data lake in modern enterprise data management. Big Data and Cognitive Computing, 6(4), 132. https://doi.org/10.3390/bdcc6040132

[19] Khan, A. Q., et al. (2022). Smart data placement for big data pipelines: An approach based on the storage-as-a-service model. In 2022 IEEE/ACM 15th International Conference on Utility and Cloud Computing (UCC) (pp. 317-320). IEEE. https://doi.org/10.1109/UCC56403.2022.00056

[20] Roman, D., et al. (2022). Big data pipelines on the computing continuum: Tapping the dark data. Computer, 55(11),74-84. https://doi.org/10.1109/MC.2022.3154148