NextGen Scientific Publication
American International Journal of Computer Science and Technology

Volume 6 Issue 1, Pg. No. 144-151, AIJCST-V6I1P114, 2024
https://doi.org/10.63282/3117-5481/AIJCST-V6I1P114

Original Article

A Scalable Distributed Data Processing Framework for Large-Scale Heterogeneous Data Integration

***Sriharsha Gudi**

Independent Researcher, Senior Data Engineer, Kirkland, Washington, USA.

Abstract:

Because a multitude of heterogeneous data is created across enterprise information systems there is skyrocketing demand for scalable and efficient distributed data processing frameworks. Computational constraints, processing delays and scalability challenges provide major limitations to traditional data integration solutions, especially dealing with large amount of structured and semi-structured data. In this paper, integration using the Apache Hadoop and Apache Spark big data technologies has been proposed which can achieve the functionality of distributing and processing the data in an upward scalable way. Proposed Architecture: An integrated Processing Architecture for the customers to batch ingest data, perform transformations, data partitioning and distributed data parallel processing. It is all about reconciling a ton of different kinds of data, and making data processing more efficient and ensuring data can be transferred reliably from one end of the enterprise data spectrum – to the other. The distributed storage system, workload distribution and resource usage and the optimization of performance for enabling data-intensive operations are explored as well. Some of these would be data partition strategies, parallelism of data and how hadoop and spark can complement each other in processing the data/computational workloads. Scalability is another important factor covered by the framework, such as data expansion, cluster sources and resources management as well as the processing bottlenecks and versatility of the infrastructure. Overall, the solution can be used to provide flexible data processing, efficiency in data processing, distributed data processing, and can be used to establish scalable data platforms for enterprises to combine various types of data.

Keywords:

Distributed Data Processing, Heterogeneous Data Integration, Apache Hadoop, Apache Spark, Batch Data Processing, Parallel Computing, Scalability Optimization.

Article History:

Received: 06.12.2023

Revised: 10.01.2024

Accepted: 24.01.2024

Published: 04.02.2024

1. Introduction

Today's hyper digital world brings together a number of new data source types from digital technologies, cloud computing, IoT (internet of things), enterprise information systems, requiring enormous amounts of data and at fast speeds. Businesses today need access to data generated by their relational databases, transactional systems, application logs, sensor networks and web services, or even semi-structured information stores. These sources often have varying data formats, storage systems, schemas, and processing needs, which can be significantly different and pose challenges for data consolidation and analysis. As data size is increasing fast, the one-center data processing system is expected to soon face problems of shortage of computation power, space etc. or even processing bottlenecks. Hence the idea of distributed data processing (DDP) has been gaining importance to consolidate several data sources as well as for enterprise scaling in analytics.

Copyright by the Author(s). This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International (CC BY-NC-ND 4.0) License (https://creativecommons.org/licenses/by-sa/4.0/)

Large-scale Heterogeneous data integration is a collection, transformation, harmonization and concatenation of the data contained in multiple, independent data sources and moving it to a uniform data processing environment. The issues that the process of integration must handle include the difference on data representation, data schema difference, operation difference (data type), data quality enforcement and even data processing reliability. In a study by Silva et al. [1] the authors investigated logical big data integration and near-real-time analytics by analyzing an approach that allows performing analytical operations across the different data sources. To them, it's important that there's "flexible integration" and "information available from one place in one access. In this direction the efficient and cost effective distributed processing, based on maximizing the proper utilization of resources, associated with workload characteristics and configuration optimization of a big data analytics platform was studied by Costa et al. [2] using data-driven techniques for performance tuning optimization. Further experiments of this kind suggest that the scalability of Data Integration is not possible with only an effective Data Coordination mechanism in place; therefore, the computation infrastructure has to be well optimized.

These are addressed with distributed architectures in which these computing tasks are broken down and distributed across multiple networked nodes where each node is able to compute independent of any others. Works by distributing data sets to nodes of the cluster and computation in parallel for large scale batch computations. Apache Spark makes this architecture possible with its in-memory computing, resilient distributed datasets and optimized execution.

Despite technologies such as distributed ones, there are a number of issues to resolve when it comes to developing scalable data integration environments of heterogeneous data. Some examples are sub-optimal Partitioning Strategy, Resource Contention, Network Communication Overhead, Format Difference in Data Sources and Irregular Data Schemas. Then there is even more: the distributed storage has to be coupled to the distributed computation the distributed processing of which you need to be co-coordinating during ingestion, transformation, execution and during output management. This must include components that process, flexible components for data ingestion, smart partitions during data ingestion and parallel processes in the computing elements.

Current work concentrates on the development of a scalable Distributed Data Processing Framework (DDPF) to process fusion and integration of such kind of data (heterogeneous) with the help of apache hadoop and apache spark. Streaming is not at the front, and the focus is on batch ingestion, structured data, transformation of semi-structured data, and coordination for distributed storage, data partitioning, parallel processing, and data performance optimization. It also explores scaling aspects that are particular to enterprise data platforms such as workload distribution, resource allocation, processing bottlenecks and infrastructure elasticity. The main aim of this is to introduce an architecture that enables efficient consolidation of data and scalable processing of it in enterprise heterogeneous environments.

2. Related Work

With increasingly large and complex enterprise datasets and need for more integrated enterprise analytical environments, there has been much research invested in scalable distributed data processing frameworks. This is because there are many related works about heterogeneous data integration, query processing in the distributed system, performance optimization, mapping the schemas and extensive processing of large amount of data. All these are significant building blocks for creating architectures that employ various distributed storage and parallel processing technologies of enterprise data integration.

To address the aforementioned issues, Silva et al. [1] suggested considering the logical integration of big data and making analytics possible in near real-time access to different data sources which could be needed in the same analysis study. Their results show that they are indeed very important in coordinating distributed information without having to consolidate all the underlying information. The problems pointed out for integration of the way are to have access to the data in various sources and a flexible execution of the queries was stressed. This research gives an adequate and timely context to build the integrated Data Processing architectures based on the application of Heterogeneous Enterprise information.

In this paper Costa et al. [2] proposed to conduct a survey, Different approaches to data-driven performance tuning of big data analytics platforms. They examined possible workload optimization as well as optimization features for system configurations, allocation of resources, and the processing performance. The point is that, with a given collection of computing hardware, appropriate configuration and workload management are needed to enable the efficiency of distributed processing.

On the basis of this study, it is concluded that as volumes increase, requirement for computational workload supported processing architectures for responsiveness to data increases too. Although it's mainly focused on anomaly detection it does demonstrate the type of processing requirements that exist in anomaly detection and can therefore be useful in designing scalable batch oriented data integration [3].

The systematic review was carried by Hajjaji et al. [4] which have done an extensive study and presented the big data and IoT application based projects used for smart environment. They were taught about the relationships between interrelated technologies, multiple sources of information and analytical applications based on data. The study highlights difficulties in governance of different kinds of data, data production using distributed devices and data processing on large-scale. The results of this demonstrate the need for mechanisms to integrate which are scalable enough to support an increasing amount of data diversity and computational needs.

Küfner et al [5] studied the vertical data continuity of lean edge analytics in Industry 4.0 production environment. What they learned was about maintaining information flow across the levels of operations and how to enable these analytical processes closer to the data creation. The study has good knowledge about why there is a need of information coordination between the two types of systems namely distributed in industry and centralized in enterprise system. These continuity demands are also relevant in the context of the heterogeneous data integration infrastructures, on which the information is produced by diverse technologies, operational processes and computation infrastructures.

The study is done pretty well covering the interoperability and query translation/query unified data access. These ideas can be used to develop distributed processing systems, in which structured and semi-structured information must be accessed and manipulated in a uniform processing fashion [6] [7].

Their work demonstrated the need of uniform query interfaces to accessing information in distributed information stored in different storage technologies. The approach tackles the challenge of handling synchronization of data access on different data repositories which have different data model and query language. This contribution is for enterprise (integration) architectures, where flexible connectivity between distributed data repositories is needed.

Sethi et al., [8] mention about Presto, which is a distributed SQL query engine to execute query over a variety of data sources. Garg and Kaur [10] studied on the integration of multiple databases where they outlined the problems related to integration of multiple databases while having information in different kinds of database environment. The research gives fundamental knowledge regarding the interoperability of database, data consolidation, data integration needs etc. Rahm and Bernstein [11] studied the automatic schema matching methods and how to find correspondences between database structures. This is applicable especially to the heterogeneous transformations, where the data are transformed from various data sources.

The fundamental notions of distributed database systems, including distributed system architecture, data fragmentation and replication, query processing, etc., were presented by Özsu and Valduriez [12]. They were involved in some of the first theory for the information management domain in both the multi-computing and/or the networked environment. Ali et al. [13] conducted survey of the schema matching between two databases with instances which refer the approach that can recognize the structural similarity and semantic relationship between two schemas present in two databases. Bonifati et al. [14] studied schema mapping transformations choosing a focus on discovering and verifying the correctness of the schema-mappings, which are necessary to exchange and integrate data[15].

3. Unified Distributed Data Fabric for Scalable Heterogeneous Data Integration

3.1. Architectural Design and Processing Layers

The framework is organized into a collection of modules or layers that each perform particular duties within the framework for performing management and computation for data. There are 6 layers of architecture: Heterogeneous data sources, Ingestion and connectivity, Distributed storage, Transformation and integration, Parallel processing and optimization and Integrated output management.

The enterprise data is from a Heterogeneous Data store Layer. It can store structured data like relational tables, transactional records and spreadsheets, as well as semi-structured data like application log files, JSON and XML. The schemas and names can be

different between data sources as well as the kinds of data and how often it is changed. Therefore, there is also a source identification mechanism provided in this framework and by using data formats, ingestion procedures and types of data could be identified.

The ingestion and connectivity layer is again a machine which is responsible for reading and/or deriving and after all transferring the data in a controlled manner from the source system into the distributed layer. Offers batch-based extraction through, connectors to databases, interfaces to file transfer systems and application programming interfaces. Data is broken up into segments according to its structure, size and manipulation. Metadata for source identification, when the source was ingested, file formats, schema characteristics etc. are maintained for transformation in the future, and for traceability.

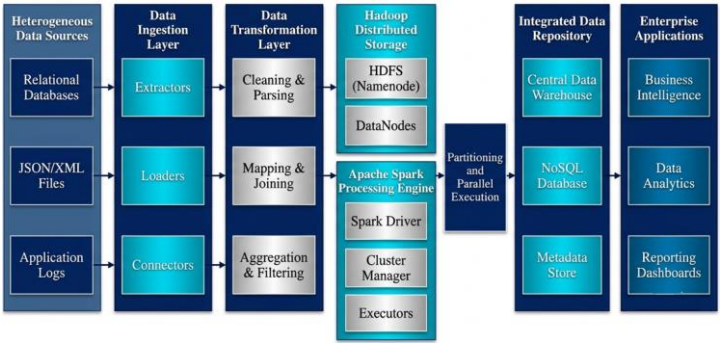


Figure 1. Overall Architecture of the Proposed Distributed Data Processing Framework

Figure 1 illustrates the overall architecture of the proposed framework, highlighting the coordinated flow of heterogeneous data through ingestion, distributed storage, parallel processing, and enterprise-level data integration.

3.2. Distributed Storage and Data Organization

A good design will result in less shuffling of the data, and will ensure that the data is easily accessible when transformed and analyzed. It's a base layer that stays in place and will not change as all the calculations are done in Apache Spark. Figure 2 depicts the distributed storage architecture, demonstrating how heterogeneous datasets are organized into distributed blocks and managed through HDFS components.

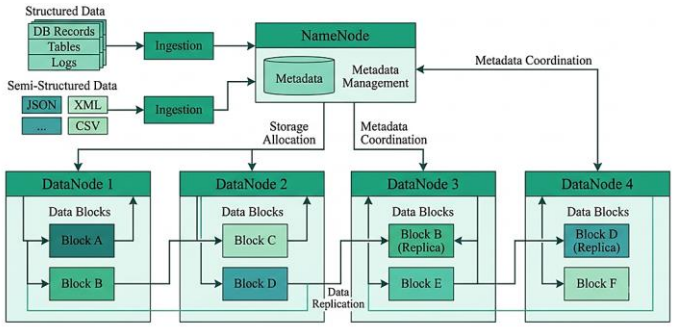


Figure 2. Distributed Storage and Data Organization Architecture

3.3. Heterogeneous Data Ingestion and Transformation Mechanism

Ingestion and transformation mechanism deals with the variations in structure, format and representation of data in enterprise sources. While batch ingestion is in progress source datasets are extracted and moved to well-defined distributed storage positions. Upon upload, the data are carefully examined to identify the different characteristics of the data, such as its format, encoding, field structure and metadata functionality.

Only duplicate fields will be detected, missing and orthogonizing the fields and the data types. Examples include date fields that have different format but should be stored in the same format, or date fields with inconsistent names that you wish to have mapped to

the same enterprise field. A "write filter" to process records from JSON and XML into an appropriate processing structure then merges with the relational records.

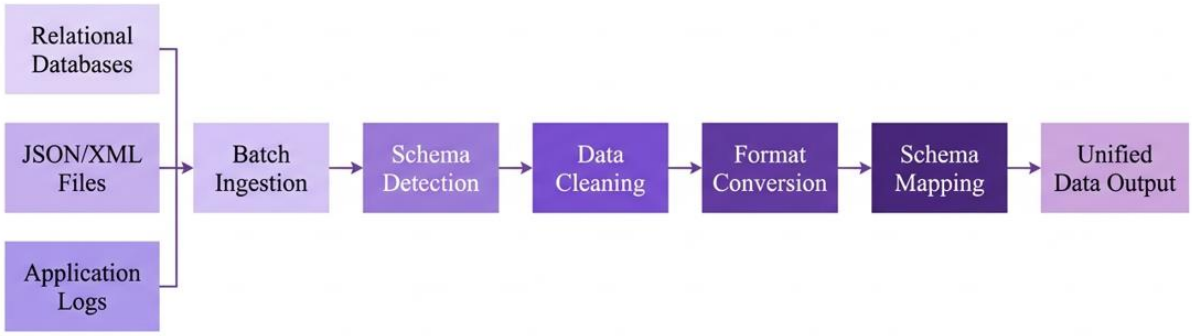


Figure 3. Heterogeneous Data Ingestion and Transformation Workflow

Figure 3 outlines the ingestion and transformation workflow, showing how diverse source formats are processed, standardized, and converted into a unified representation for distributed analysis. Relationships between the source attributes and target field (standardized) are created by a logical mapping mechanism.

The transformations occur via distributed operations over the data, all of which take place behind the scenes of Apache Spark. The flexible data processing capabilities with semi-structured data are complemented with its structured transformation facilities both DataFrame and SQL interfaces.

3.4. Intelligent Partitioning and Parallel Execution

For structured data, partitioning can be done on user-defined attributes like geographic keys, business keys or transaction dates. Filing organization may make a semi-structured judgment on information regarding the source identifier(s), the ingestion period or even on the filing group. The size of a partition is important as small partitions result in high scheduling overhead, but very large partitions can lead to an imbalance in processing and underutilization of resources.

Transformation/Integration work is done within distributed partitions using Apache Spark. It has an execution engine, which is used for task scheduling, sharing data and intermediate calculation. Multiple sections are able to execute compatible operations simultaneously with parallel processing thus minimizing the sequential dependency. Figure 4 presents the partitioning and parallel execution model, emphasizing the distribution of data segments across worker nodes and the coordinated aggregation of processing results.

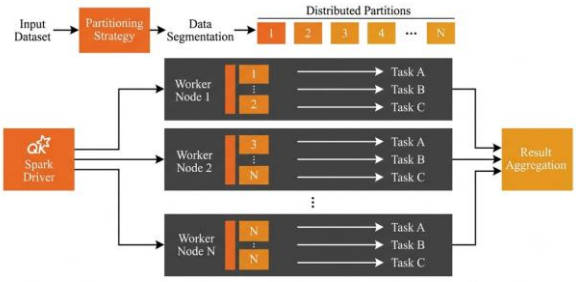


Figure 4. Intelligent Data Partitioning and Parallel Execution Model

This framework also takes into account the locality of data, meaning that operations are performed locally near the nodes that hold the data. This will prevent network communication from being on start-up and utilize resources efficiently. Partitions can be redistributed and workloads balanced, thus reducing any data skews or workload imbalances.

3.5. Hadoop–Spark Coordination and Performance Optimization

Finally, as part of the proposed model, both Hadoop and Spark are used.

The coordination mechanism takes care of indicating the availability of the datasets, requesting their processing, providing resources and indicating the status of the execution as well. We can leverage hadoop YARN to manage resources, and to sync the computational resources in the cluster. Applications are built on Spark that load datasets distributed in HDFS and execute processing and use the resources allocated in the cluster.

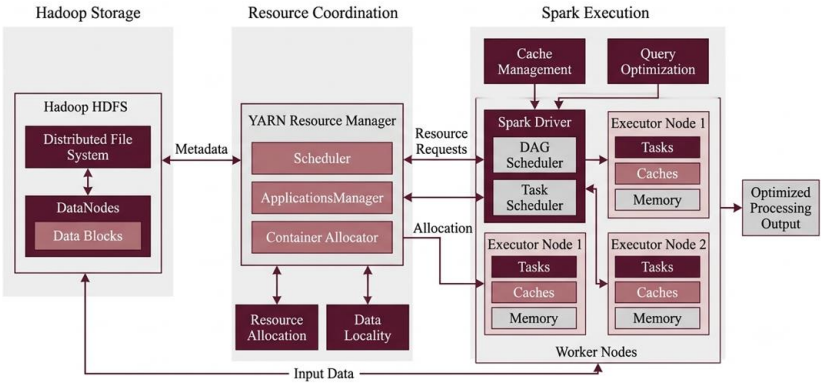


Figure 5. Hadoop–Spark Coordination and Performance Optimization Framework

Figure 5 demonstrates the coordination between Hadoop and Spark, highlighting resource allocation, data locality, caching, and query optimization within the distributed processing environment.

Several operational properties used for performance optimization include the size of the partition, how parallel tasks are distributed, memory resources, data serialization and data exchange over the network. Intermediate data sets that are used well, have the potential of being buffered on the right cache to reduce unnecessary access to the cache and data sets. Queries are efficiently optimized and efficient join strategies minimize data shuffling. When allocating resources, contention in computation and memory resources are also taken into consideration to avoid over usage.

3.6. Integrated Output Management and Scalability Governance

The end layer is for transformed and integrated data for downstream enterprise applications, reporting systems, business intelligence systems and analytical repositories. All the information processed is stored in the predetermined schemas and is stored at predetermined locations. For traceability and later use, the assumption is that the metadata of the datasets have been recorded, such as processing dates and/or times, transformation status and sources.

Scalability governance is the ability to support an expansion in the volume of data, the number of data sources and compute requirements. They can be extended horizontally (more nodes for storage and/or processing) and they are able to distribute the work across nodes. The operational reliability, controlled access to the data, processing logs and data validation, and metadata consistency support operational reliability and controlled data handling.

Source identification, batch ingestion, distributed storage, schema harmonization, data transformation, partition allocation, parallel processing, optimization for performance and integration output are all well synchronized aspects of the above processing workflow. The entire stages collectively result in an architecture that can be used to reuse for enterprise data integration.

In conclusion, the main issues addressed in the presented architecture are integration of large scale and heterogeneous data which is achieved in the architecture through distributed storage solution (Hadoop) and computational processing solution (spark). It can be organized into layers, can be transformed using metadata, is executed based on partitions and can be optimized for resources. The framework includes scalability, processing coordination and consistency of integration in a single distributed system and can be scaled to structured and semi-structured information.

4. Empirical Assessment of Framework Capabilities and Enterprise Scalability

The data processing aspects of this proposed distributed data processing are quantified and qualified through an empirical architecture oriented data processing analysis in order to quantify and qualify processing capacities, integration mechanisms, and operational characteristic and scalability aspects. It concentrates on the issues on both ends of the spectrum with regards to heterogeneous, large scale data integration and the benefits that Apache Hadoop together with Apache Spark offer. It is not a matter of experimental measurement or benchmarking, but one of suitability of the functions for the individual architectural component(s) and of enterprise data processing needs.

4.1. Enterprise Scalability and Operational Adaptability

Scalability of the framework in relation to the quantity of input data sources, the computation needed for the data, and the amount of data is discussed. Furthermore, the metadata management, operational data traceability and data processing validation ensure an operational consistency. Thus, the distributed architecture of the framework is appropriate for enterprise scenarios in which flexible data integration infrastructures have to be provided.

4.2. Distributed Processing and Computational Coordination

The coordination, persistence data and computation performed in another entity can be distinguished in their context thanks to the coordination. It comes in handy to prevent the data and computations from becoming pointy, when dealing with a lot of data, distributed on different machines. Furthermore, distributed task execution can be used to subdivide the workloads, so that the distinct partition of data will be worked out on different cluster nodes.

4.3. Partitioning Efficiency and Workload Management

Partition oriented processing provides flexibility in structuring large data on the basis of business attributes or on the basis of source characteristics or on the basis of ingestion time. Using appropriate partitions, unwanted access to data can be reduced and also parallelism can be improved. Also the workload balancing aspect must be taken into account which sets up a limit to the extent to which the workload can be apportioned and also stores record distribution and computation complexity. Data uniformity and efficient redistribution of the data partitions as well as the efficient join operations minimize the communication overhead. These mechanisms provide a framework as to how workload can be handled in a computational environment and as little as possible of the shared resource can be used in a distributed environment.

4.4. Performance Optimization and Resource Utilization

Various optimizations have been adopted and discussed in detail in the following chapters including memory management, intermediate data caching and query optimization, serialization and distribution of resources. If, when transforming a data set, Spark needs to do multiple stages of processing, it stores the intermediate result in memory so that unnecessary access to this data on disk is avoided. As of this writing, however, in a distributed cluster environment, Hadoop Yarn is the resource coordinating component.

5. Conclusion

Nevertheless, its performance in different deployment scenarios has to be validated by real deployment and systematic performance analysis. Its future improvements with AI-Driven Orchestration, Real-Time Stream Processing, Automated Schema Evolution, Cloud-Native Deployments, and Advanced Governance would add even greater power. In all, it provides a broad basis for specifying distributed data-processing environments that can be flexible, interoperable, scalable and that address many different and changing enterprise data-analysis and data-integration needs.

References

[1] Silva, B., Moreira, J., & Costa, R. L. C. (2023). Logical big data integration and near real-time data analytics. Data & Knowledge Engineering, 146, 102185. https://doi.org/10.1016/j.datak.2023.102185

[2] Costa, R. L. de C., Moreira, J., Pintor, P., dos Santos, V., & Lifschitz, S. (2021). A survey on data-driven performance tuning for big data analytics platforms. Big Data Research, 25, 100206. https://doi.org/10.1016/j.bdr.2021.100206

[3] Habeeb, R. A. A., Nasaruddin, F., Gani, A., Targio Hashem, I. A., Ahmed, E., & Imran, M. (2019). Real-time big data processing for anomaly detection: A survey. International Journal of Information Management, 45, 289–307. https://doi.org/10.1016/j.ijinfomgt.2018.08.006

[4] Hajjaji, Y., Boulila, W., Farah, I. R., Romdhani, I., & Hussain, A. (2021). Big data and IoT-based applications in smart environments: A systematic review. Computer Science Review, 39, 100318. https://doi.org/10.1016/j.cosrev.2020.100318

[5] Küfner, T., Schönig, S., Jasinski, R., & Ermer, A. (2021). Vertical data continuity with lean edge analytics for Industry 4.0 production. Computers in Industry, 125, 103389. https://doi.org/10.1016/j.compind.2020.103389

[6] Tan, R., Chirkova, R., Gadepally, V., & Mattson, T. G. (2017). Enabling query processing across heterogeneous data models: A survey. In 2017 IEEE International Conference on Big Data (Big Data) (pp. 3211–3220). IEEE. https://doi.org/10.1109/BigData.2017.8258302

[7] Kolev, B., Valduriez, P., Bondiombouy, C., Jiménez-Peris, R., Pau, R., & Pereira, J. (2016). CloudMdsQL: Querying heterogeneous cloud data stores with a common language. Distributed and Parallel Databases, 34(4), 463–503. https://doi.org/10.1007/s10619-015-7185-y

[8] Sethi, R., Traverso, M., Sundstrom, D., Phillips, D., Xie, W., Sun, Y., Yegitbasi, N., Jin, H., Hwang, E., Shingte, N., & Berner, C. (2019). Presto: SQL on everything. In 2019 IEEE 35th International Conference on Data Engineering (ICDE) (pp. 1802–1813). IEEE. https://doi.org/10.1109/ICDE.2019.00196

[9] Silva, B., Moreira, J., & Costa, R. L. C. (2021). EasyBDI: Near real-time data analytics over heterogeneous data sources. In Proceedings of the 24th International Conference on Extending Database Technology (EDBT) (pp. 702–705). https://doi.org/10.5441/002/edbt.2021.88

[10] Garg, B., & Kaur, K. (2015). Integration of heterogeneous databases. In 2015 International Conference on Advances in Computer Engineering and Applications (pp. 1033–1038). IEEE. https://doi.org/10.1109/ICACEA.2015.7164859

[11] Rahm, E., & Bernstein, P. A. (2001). A survey of approaches to automatic schema matching. The VLDB Journal, 10, 334–350. https://doi.org/10.1007/s007780100057

[12] Özsu, M. T., & Valduriez, P. (2020). Principles of distributed database systems (4th ed.). Springer. https://doi.org/10.1007/978-3-030-26253-2

[13] Ali, A., Nordin, A., Alzeber, M., & Zaid, A. (2017). A survey of schema matching research using database schemas and instances. International Journal of Advanced Computer Science and Applications, 8(10). https://doi.org/10.14569/IJACSA.2017.081014

[14] Bonifati, A., Mecca, G., Papotti, P., & Velegrakis, Y. (2011). Discovery and correctness of schema mapping transformations. In Schema matching and mapping (pp. 111–147). Springer. https://doi.org/10.1007/978-3-642-16518-4_5

[15] Abadi, D., Ailamaki, A., Andersen, D., Bailis, P., Balazinska, M., Bernstein, P., Boncz, P., Chaudhuri, S., Cheung, A., Doan, A., et al. (2020). The Seattle report on database research. ACM SIGMOD Record, 48(4), 44–53.