Report for the Degree of Master of Computer Science

End to End Automation in ETL Pipeline

Pikesh Maharjan

(LC00015002358)

Masters in Computer Science (MCS) IIMS college

Computer Science Department

Lincoln University, Malaysia

March, 2023

Research project for the Degree of Master of Computer Science

End to End Automation in ETL Pipeline

Supervised by Prof. Sudan Jha, Ph.D.

A report submitted in partial fulfilment of the requirements for the degree of Master of Computer Science

Pikesh Maharjan

(LC00015002358)

Masters in Computer Science (MCS) IIMS college

Computer Science Department

Lincoln University, Malaysia

March, 2023

Table of Contents

CHAPTER 1: INTRODUCTION

1.1 Background

ETL (Extract, Transform, Load) pipelines play a vital role in today's world where every second huge number of data is being generated. It plays a key role in the field of data management and analytics by enabling movement and transformation of raw data and then loading in any databases, data warehouses or any other storage systems. It can then be converted into actionable insights which are very important in process of decision making in any data-driven organization. However, traditional ETL workflows are often manual where human intervention is required in various processes such as schema detection, data type categorizing, mappings and other processes which makes it time-consuming. Manual ETL process can lead to inconsistencies, issues and delays which can impact the correct and timely generation of required information from the data. With the exponential growth of volume and large variety of data, It is a need to automate the ETL pipelines where less numbers of human effort is required. Automating ETL pipelines can handle large amount of data with large numbers of fields with minimum human efforts, reducing the time delays and issues that could arise due to human errors. Thus, automating ETL pipelines has become a necessity to improve efficiency, scalability, and reliability.

1.2 Statement of the Problem

The manual execution of Extract, Transform, Load (ETL) processes has been a persistent challenge in data-driven industries with the growing amount of data. Traditional ETL workflows often require extensive human intervention to handle tasks such as file ingestion, Meta Data recognition, mapping fields from source to destinations, and transformation logic definition, leading to inefficiencies in time and resource management. These workflows are particularly prone to errors when dealing with diverse data formats and varying levels of data quality. Furthermore, manual pipelines struggle to scale with the increasing volume of big data, creating bottlenecks in critical data operations. Identifying data types accurately in heterogeneous datasets remains a challenge, with rule-based methods lacking adaptability and requiring significant maintenance effort. Machine learning approaches, while promising, are underutilized in this domain due to their perceived complexity.

Another critical issue lies in the lack of dynamic configuration systems that allow seamless integration of user-defined transformation logic. Current practices often require modifying pipeline code for every new dataset, which is not only time-consuming but also limits

flexibility. Additionally, there is a gap in automating the final loading stage into target systems like SQL Server, which often involves repeated manual validations and corrections. These limitations not only delay decision-making but also increase the likelihood of data inconsistencies, making it imperative to explore automation-driven solutions. This research seeks to address these challenges by proposing a unified framework that integrates machine learning and rule-based approaches for data type identification, a dynamic configuration system for user-driven transformations, and an automated pipeline for seamless loading into SQL Server. By addressing these gaps, the study aims to improve the efficiency, scalability, and accuracy of ETL processes, paving the way for more robust data management systems.

1.3 Research Questions

The study seeks to address the following questions:

1. How can machine learning methods be integrated to identify data types in ETL pipelines?
2. What are the optimal approaches to create human-readable configuration files?
3. What optimizations can be introduced to improve the speed and efficiency of data loading without compromising data integrity?

1.4 Objectives

General Objective:

1. To design and implement an automated ETL pipeline capable of handling diverse datasets while ensuring accuracy and scalability.

Specific Objectives:

1. Develop mechanisms for data type recognition using ML methods.
2. Create a dynamic configuration file to allow human intervention when necessary.
3. Automate the data loading process into SQL Server, including transformation and validation steps.

1.5 Significance of the Study

The significance of this study lies in its potential to change the traditional way of data engineering by automating the end-to-end ETL pipeline. This research will minimize manual interventions in data processing, reducing human error and accelerating data integration for decision-making processes. It will provide a robust framework for handling delimited file

formats, performing data type identification, and implementing configurable transformation logic, which can be directly applied in real-world scenarios. Furthermore, this study will contribute to the scalability and efficiency of ETL operations, enabling organizations to handle large volumes of data with improved reliability. The outcomes of this research will benefit data engineers, analysts, and businesses by streamlining workflows and enhancing the overall quality and accessibility of data.

1.6 Scope and Limitations of the Study

The scope of this research focuses on designing and implementing an automated end-to-end ETL (Extract, Transform, Load) pipeline. It encompasses the development of a front-end interface for file uploads, APIs for data ingestion, and a system to identify data types with help of machine learning approaches. Additionally, the study includes the creation of a configuration file to define metadata such as headers, delimiters, and transformations, along with functionality to enable human intervention for validation or modification of the configuration. The scope extends to loading processed data into a SQL Server database and ensuring scalability and adaptability for handling Delimited file format.

The study will primarily focus on structured file format (Delimited files), with limited exploration of semi-structured or unstructured data formats. The research will not include real-time data ingestion and processing, as the pipeline is designed for batch processing scenarios. The research will use synthetic datasets, avoiding sensitive or proprietary data, which may impact the generalization of results in industry-specific contexts.

CHAPTER 2: LITERATURE REVIEW

A literature review surveys the existing body of knowledge relevant to the automation of ETL (Extract, Transform, Load) pipelines. This process is critical for understanding theoretical frameworks, methodologies, and practical applications discussed in scholarly work. Numerous studies have explored the challenges and advancements in automating data integration and transformation processes, emphasizing the growing importance of scalable and efficient ETL systems in data-intensive industries.

For example, researchers have highlighted the use of rule-based and machine-learning approaches to infer schema and data types from semi-structured and structured data sources. These studies reveal that while machine learning provides higher accuracy in complex datasets, rule-based methods are computationally lighter and more interpretable, making them suitable for initial configurations.

The development of user-configurable metadata files, such as configuration files specifying headers, delimiters, and transformations, has also been discussed extensively in the literature. Such systems allow human interaction to validate and modify the pipeline's assumptions, balancing automation with flexibility. However, studies also point out the need for robust mechanisms to handle anomalies and ensure data quality during transformation processes.

Previous research has proposed the integration of SQL-based database systems for loading transformed data, with many emphasizing the use of SQL Server for its robust support for batch data processing and reporting capabilities. Additionally, best practices in ETL pipeline design, including modularity, scalability, and error handling, have been extensively documented.

In summary, the literature underscores the importance of designing ETL pipelines that are not only automated but also adaptive, efficient, and user-friendly. This review identifies gaps in practical implementations, such as handling dynamic configurations and optimizing for scalability, which this research aims to address.

CAHAPTER 3: METHODOLOGY

The methodology for automating the ETL pipeline comprises several phases, beginning with data ingestion and ending with the loading of transformed data into the target database. This structured approach ensures a seamless flow of data while allowing room for human intervention where necessary. Each stage is designed to address specific aspects of the ETL process efficiently and effectively.

Data Ingestion

The process starts with a web-based frontend where users upload the file to be processed. An API is then developed to fetch this uploaded file and transfer it to the backend. The API handles various delimited file format, ensuring flexibility in handling diverse data sources. The backend validates the file to ensure compatibility and integrity before proceeding to the next stage.

Data Type Identification

To identify data types, a combination of rule-based logic and machine learning techniques is employed. The rule-based system examines predefined patterns (e.g., date formats or numerical values) to infer column data types. Simultaneously, machine learning models are trained on sample datasets to predict data types with higher accuracy, particularly for ambiguous cases. This dual approach enhances reliability and adaptability in detecting column types.

Configuration File Generation

Once the data types are identified, a configuration file is automatically generated. This file provides detailed metadata about the dataset, including column names, data types, delimiters, and record separators. The configuration file acts as a blueprint for subsequent transformations and loading operations.

Human Interaction for Validation

A provision is made for human users to review and modify the generated configuration file. This optional step is controlled via a check button on the frontend interface. If enabled, users can view the configuration file and adjust, such as adding specific transformation logic or

correcting any discrepancies in data type detection. This step ensures greater flexibility and allows customization for unique use cases.

Incorporating Transformation Logic

If additional transformation rules are required, users can specify them directly in the configuration file. For example, a user may need to standardize date formats, remove invalid rows, or aggregate data based on specific conditions. These transformations are dynamically parsed and applied to the dataset, ensuring that the data meets the desired quality and format standards before loading.

Data Transformation and Cleaning

Using the instructions from the configuration file, the backend applies all specified transformations. These include renaming columns, handling missing values, normalizing data, and applying advanced transformations as defined by the user. This phase ensures that the dataset is prepared for seamless integration into the target database.

Data Loading

The final step involves loading the transformed data into a SQL Server database. The loading process uses optimized techniques to handle large datasets efficiently. Indexing and batch operations are employed to minimize load time and enhance performance. The system also logs the loading process, capturing details such as the number of rows loaded, errors encountered, and time taken.

Automation and Scalability

Although workflow orchestration tools like Apache Airflow are not implemented in the initial version, provisions are made to include automation for scheduling and monitoring tasks in future iterations. The current process can be run manually or triggered programmatically, ensuring flexibility in deployment.

CHAPTER 4: REFERENCE

[1] K. C. Mondal, N. Biswas, and S. Saha, "Role of Machine Learning in ETL Automation," in *Proceedings of the 21st International Conference on Distributed Computing and Networking (ICDCN 2020)*, Kolkata, India, Jan. 2020, pp. 1-6. Available: (PDF) Role of Machine Learning in ETL Automation

[2] W. Yaddow, "Considerations for Automating Data Warehousing and ETL Tests," presented at the Datagaps, Feb. 2020. Available: (PDF) Considerations for Automating Data Warehousing and ETL Tests

[3] C. Van der Putten, "Transforming Data Flow: Generative AI in ETL Pipeline Automatization," M.S. thesis, Dept. Data Sci. and Eng., Politecnico di Torino, Turin, Italy, Apr. 2024. Available: tesi.pdf

[4] P. Pham, "A Case Study in Developing an Automated ETL Solution – Concept and Implementation," Bachelor's thesis, Dept. Inf. and Commun. Technol., Turku Univ. of Appl. Sci., Turku, Finland, 2020. Available: https://www.theseus.fi/bitstream/handle/10024/340208/Pham_Phuong.pdf?sequence=2

[5] M. T. Maulik, "Automated ML ETL Pipeline of Electric Motor Temperature Sensor Data for Commercial Vehicles," Master's thesis, Dept. Data Sci. and Eng., Politecnico di Torino, Turin, Italy, Apr. 2024. Available: https://github.com/maulikt04/Automated-ML-ETL-pipeline-of-electric-motor-temperature-sensor-data-for-commercial-vehicles-.