# Abstract

Description: In the data science industry, esp. in the advent of large volumes of data and data processing systems, the security and reliability of the data is a serious concern. Understanding data lineage enables auditing and accountability and allows users to better understand the flow of data through a system.

The approaches tried so far for tracing data lineage and its challenges are:

* Certain heuristics or code analysis - which are platform dependent, built for specific languages like SQL or specific to Data processing technology like spark.
* Tools like Databricks, Airflow, and DBT come with lineage and tagging features that work just fine. These work fine in isolation till one has to deal with multiple systems ( heterogenous of systems).
* Word embeddings directly into the data to trace lineage - which is inefficient because of the large storage requirements and lack of practicality of implementing them into existing systems.
* Completely relying on the data to find Lineage - ML based solutions use traditional ML techniques like classification, regression, clustering etc. done one the raw data directly to extract lineage. This might work but isn’t practically feasible.

The aim of this research is to leverage the metadata and transformation logic along with techniques from NLP/LLM resulting in effective data lineage discovery. Since we rely on NLP the solution is language/platform independent making it applicable to wider range of data pipelines.

**Table of Contents**

[Abstract 1](#_Toc75518344)

[1. Background 3](#_Toc75518345)

[2. Problem Statement OR Related Research OR Related Work 3](#_Toc75518346)

[3. Research Questions (If any) 3](#_Toc75518347)

[4. Aim and Objectives 3](#_Toc75518348)

[5. Significance of the Study 4](#_Toc75518349)

[6. Scope of the Study 4](#_Toc75518350)

[7. Research Methodology 4](#_Toc75518351)

[8. Requirements Resources 4](#_Toc75518352)

[9. Research Plan 4](#_Toc75518353)

[References 4](#_Toc75518354)

# Background

The definition of Data Lineage is its entire processing history including its origin (e.g., the

identifier of the base data set, the recording instrument, the instrument's operating parameters) as well as all subsequent processing steps (algorithms and respective parameters) applied to it. (Supporting Fine-Grained Data Lineage in a Database Visualization Environment - Allison Woodruff and Michael Stonebraker)

Benefits of lineage to various industries (especially finance industry)

1. Automated impact analysis

2. Increased data pipeline observability

3. Regulatory compliance

4. Efficient cloud migrations

5. Improved workflow & IT retention

6. Trust and data governance

Large volume of data and advent of cloud has made it difficult to keep track of lineage. (Data Provenance and Data Lineage in the Cloud: A Survey). Considering these data processing pipelines are heterogenous as they use different tools and techniques for data transformation, each system works in isolation making tracing lineage further more difficult. Also, with such large data volume the security and reliability of the data is a serious concern.

# Solutions to Data Lineage problems

Data Lineage can be obtained by the following processes:

* Manually implemented lineage
* Pattern-based lineage
* Lineage through data tagging
* Data lineage by parsing

Out of these, by far Lineage by parsing is the best solution as it offers a practicality and wider applicability. However parsing queries is very complex, varies according to query language used and traditional query parsing algorithms are generally very specific to a particular Query language.

# Machine Learning based solutions to Data Lineage

Traditional data lineage solutions are based on heuristics and come coupled with tools and techniques of the particular pipeline, like SQL or specific to Data processing technologies like spark making them limited in scope. Thus, a Machine Learning approach would be preferable.

The approaches tried so far:

- Certain heuristics or code analysis

- Word Embeddings directly into the data to trace lineage: These are inefficient because of the large storage requirements and lack of practicality of implementing them into existing systems. [ML Based Lineage in Databases, Michael Leybovich, Oded Shmueli]

- Completely relying on the data to find Lineage: ML based solutions use traditional ML techniques like classification, regression, clustering etc. or advanced techniques like NLP.

[Tracer: A Machine Learning Approach to Data Lineage by Felipe Alex Hofmann] [Tracer: A Machine Learning Based Data Lineage Solver with Visualized Metadata Management by Zhuofan Xie]

# 3. Research Questions (If any)

* Is an effective Machine Learning based solution possible for Data Lineage problems?
* Can that be made generic enough?
* If yes, can ML based solution work be as good as traditional DL methods even when generic?

# 4. Aim and Objectives

**Aim:**

To find a Machine learning solution to data provenance, thereby solving data lineage problem.

**Objective:**

1. Using NLP to parse the logic in transformation at every granular stage and perform Column Map discovery.
2. Stitch back multiple Column Maps discovered at various stages to trace the data lineage.

# 5. Significance of the Study

In data science, the inability to trace the lineage of particular data is a common and fundamental problem that undermines both security and reliability [Patrick McDaniel. Data provenance and security. IEEE Security & Privacy, 9(2):83–85, 2011.]. To ensure users easily understand how the data flows through the system and to ensure accountability, it is important that we trace Data Lineage. But a common problem is that by the time data passes through the system, the lineage is lost. Here we try to build a machine learning solution to the data lineage problem.

* An ML based approach would ensure secure and reliable data lineage tracing.
* It will be a generic solution removing the major problem of limited scope.
* This approach will be more practical and resource efficient than the previously tried ML approaches.

The aim of this research is to leverage both contents of the data and the metadata along with techniques from NLP/LLM resulting in effective data lineage discovery.

# 6. Scope of the Study

An end-to-end solution for a Data lineage problem would involve the following:

1. Discovering lineage information at granular levels of data pipeline - The stage of obtaining lineage by parsing and traversing transformation queries.
2. Collecting and storing this lineage information at various levels – A graph problem involving building graph structures to effectively store the obtained lineage info at the granular levels.
3. Aggregating/Stitching back the collected lineage information to provide useful information.

The scope of this research is to find data lineage at a granular level of the data transformation. The other two parts are not touched upon in this research, since the other two parts are already addressed by prevalent graph techniques.

# 

# 7. Research Methodology

Creating Data pipeline Simulation

**Data**

We consider the tabular data as the basis for this study as it the most generic form of data available. It is this form that is mostly used in the regular data pipelines in the industry such as finance domain etc.

**Dataset** - multi-table relational datasets ( <https://arxiv.org/abs/1511.03086> )

Data Volume: ~2GB in size.

This repository contains 73 multi-table relational datasets covering a diverse range of applications from Credit data to Airline Industry. It includes standard benchmark datasets used for research on data processing systems. Each dataset consists of multiple tables which are related through various primary and foreign key relationships.

**Experimental Setup**

Build a data pipeline simulation by using various types of transformation queries. These set of queries used for data transformations needs to represent various aspects of a typical data pipeline.

Use transformation queries from the collected reference to generate derived columns for each table. Persist the expanded tables and updated metadata. Load the tables and metadata and generate data pipeline establishing multiple levels of transformations persisted into the newly derived tables.

Tables

Datasets

Tb1

Transformation Query (ex. SQL - DML)

Tbn”

Datasets

Tbn

Datasets

A typical transformation query has the following structure (**Basic parts** of a query)\*

* **projections** - ex. select clause in SQL
* **relations** - ex. from clause in SQL
* **filters** (Slicing and dicing) - ex. where clause expressions in SQL

There are other items in the above list of structure such as order and grouping constructs.

But are not primary drivers of lineage.

Assuming simple query (without any nesting or hierarchy - for the sake of simplicity). As for once the solution is established it can be easily be extracted to handle more complex scenarios by applying the same techniques recursively and iteratively.

Model building

Similar to Tracer app (Xie-zhuofanx) we will be focusing on the Column map discovery but using the transformation logic in the query.

**Column map** definition\* - The mapping between a target table column to the columns from the parent table that generated it. (Data Structure: a sorted list of dictionaries.)

High level stages of **parsing** the transformation query:

1. Split into parts – **documents\*** oftype **projections**, **relations** and **filters**.
2. Extract topics (column names) **Classify\*** them in to Source and target table attributes,
3. **Cluster\*** them based on relative distance to obtain relationships.

Transformation query is fed into a language processing pipeline as text (tokenizer, tagger, parser, ner etc.), which will be built as part of this study, to split the query into its basic parts (projections, relations and filters) being collected into its set of documents.

The documents extracted will be used as training data for the model. Key Value pairs of input columns vs output columns which will be manually created for training and test data. We shall use NLP algorithms such as Topic modelling, LDA etc. to derive the input/output column names from the previously created documents.

Generate these key Value pairs which we shall later refer to as the column maps using clustering mechanisms such as KNN.

**Proposed Solution**

<column names 1…n>

Transformation Query (ex. SQL - DML)

Classification

Topic Modelling

select

from

Model evaluation.

where

<source column names 1…n>,

<target column names 1…n>

Clustering

Column map

{Source column1: <column names 1…n>} …

Model evaluation

Performance evaluation is done to select the best model/ combination of models at every stage of query parsing. Tuning and improvement is done to improve performance. And then finally it is compared with reference implementations for drafting conclusions.

The model is evaluated for **precision** and **accuracy**.

# 8. Requirements Resources

Dataset used: Multi-table relational datasets - https://arxiv.org/abs/1511.03086

The tests have been performed on a PC with the subsequent specifications:

CPU: Intel Core i7, 2.8 GHz

RAM: 6 Gigabytes

OS: Windows 10 Enterprise

Software: IntelliJ 8.2, Jupyter notebook to run the data pipeline simulation.

# 

# 9. Research Plan

1. Setup (2 weeks)

- Data collection and structuring to my needs.

- establishing Sample Data transformation pipeline.

- Incorporate various warehouse schematics.

2. ML Model Evolution. (4 weeks + 2 buffer)

- Setup NLP preprocessing pipeline for training and test data.

- Performance evaluation

- Performance comparison with references.

- Fallback approach (if required) using LLM to achieve the same results.

3. Risks or contingency plan. (2 weeks)

- If NLP fails to perform. Use alternative and relatively easy implementation option such as LLMs to generate the similar outcome.

4. Documentation and report submission (2 weeks)

# 

# References

* Azhir, E., Jafari Navimipour, N., Hosseinzadeh, M., Sharifi, A., & Darwesh, A. (2021). An automatic clustering technique for query plan recommendation. Information Sciences, 545, 620–632. https://doi.org/10.1016/j.ins.2020.09.037
* Bertino, E., Bhattacharya, S., Ferrari, E., & Milojicic, D. (2023). Trustworthy AI and Data Lineage. In IEEE Internet Computing (Vol. 27, Issue 6, pp. 5–6). Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/MIC.2023.3326637
* Graefe, G., & Ward, K. (n.d.). Dynamic Query Evaluation Plans.
* Graves, L., Nagisetty, V., & Ganesh, V. (2021). Amnesiac Machine Learning. 35th AAAI Conference on Artificial Intelligence, AAAI 2021, 13A, 11516–11524. https://doi.org/10.1609/aaai.v35i13.17371
* Hofmann, F. A. (2020). Tracer: A Machine Learning Approach to Data Lineage.
* Leybovich, M., & Shmueli, O. (2021). ML Based Lineage in Databases. http://arxiv.org/abs/2109.06339
* Mahadevan, A., & Mathioudakis, M. (2022). Certifiable Unlearning Pipelines for Logistic Regression: An Experimental Study. Machine Learning and Knowledge Extraction, 4(3), 591–620. https://doi.org/10.3390/make4030028
* Patel, S., Rahevar, M. L., Parmar, M., Patel, P. U., & Rahevar, M. (2020). Data Provenance and Data Lineage in the Cloud: A Survey. International Journal of Advanced Science and Technology, 29(5), 4883–4900. https://www.researchgate.net/publication/344188496
* Tang, M., Shao, S., Yang, W., Liang, Y., Yu, Y., Saha, B., & Hyun, D. (2019). SAC: A system for big data lineage tracking. Proceedings - International Conference on Data Engineering, 2019-April, 1964–1967. https://doi.org/10.1109/ICDE.2019.00215
* Woodruff, A., & Stonebraker, M. (n.d.). Supporting Fine-Grained Data Lineage in a Database Visualizration Environment.
* Xie, Z. (2022). Tracer: A Machine Learning Based Data Lineage Solver with Visualized Metadata Management.
* Zhang, Z., Sparks, E. R., & Franklin, M. J. (2017). Diagnosing machine learning pipelines with fine-grained lineage. HPDC 2017 - Proceedings of the 26th International Symposium on High-Performance Parallel and Distributed Computing, 143–153. https://doi.org/10.1145/3078597.3078603
* Zhao, Y., Li, Z., & Cong, G. (2023). A Comparative Study and Component Analysis of Query Plan Representation Techniques in ML4DB Studies. Proceedings of the VLDB Endowment, 17(4), 823–835. <https://doi.org/10.14778/3636218.3636235>
* https://www.ibm.com/blog/6-benefits-of-data-lineage-for-financial-services/