Generator of Multi-Model Data

Project Report - Outline of Development Steps and Decisions

Alžběta Šrůtková

November 28, 2024

Charles University
Software and Data Engineering Department
Prague, Czech Republic

Table of Contents

1 Introduction **1**

2 Project Description **1**

3 Project Execution **2**
3.1 Understanding the Tools . . . 2
3.2 Integration . . . 2
3.2.1 Integration of MM-infer into MM-cat . . . 2
3.2.2 Conversion Component . . . 3
3.3 Other implementation work . . . 3
3.3.1 Creation of new Wrappers . . . 3
3.3.2 Unification of Data Sources . . . 3
3.3.3 Integration and Adjustment of the Candidate Miner Package . . . 4
3.3.4 Development of the Final Pipeline . . . 4
3.3.5 Improving user experience . . . 5
Editor for Inferred Schema Category . . . 5
Candidates . . . 6
Layout . . . 7
Mapping Editor . . . 7
Hierarchical Task Analysis . . . 7
3.3.6 Constant Adaptation to Ongoing Code Development . . . 8
3.4 Testing and Validation . . . 9
3.5 Dataset Exploration . . . 9
3.5.1 Criteria for Dataset Selection . . . 10
3.5.2 Real-world Datasets . . . 11
Yelp . . . 11
IMDb . . . 11
Housing in London Boroughs . . . 12
BibleData . . . 12
NASA . . . 12
LDBC SNB Social Network Data . . . 12
SWAPI . . . 13

4 Next Steps and Dropped Concepts **13**
4.1 Ideas with Potential . . . 13
4.2 Discarded Ideas . . . 14

5 Conclusion **14**

1 Introduction

This document provides a comprehensive overview of the project's development process. It highlights the various challenges encountered and the strategies employed to address them. Due to the unique nature of the project, it became necessary to create this additional document to capture aspects of the process that were not fully covered in the existing documentation.

It aims to provide a more complete picture of the project's scope and complexity. And offers readers an understanding of the diverse range of tasks involved, from integrating complex tools and technologies to navigating the complexities of dataset selection and the final pipeline creation.

2 Project Description

The full description of the project is available in the Project Specification document. Here, I will provide a brief overview of the project's main goals and objectives.

The primary goal of this research project is to develop a tool capable of generating pseudo-real multi-model data from any given input single-model dataset. This tool will not be developed entirely from scratch; instead, it will be built by integrating several existing tools that were developed in previous theses and projects supervised by members of our research team. These tools include:

- MM-cat [12] - a tool for conceptual modeling of multi-model data
- MM-infer [15] - a tool for inference of a conceptual model from a set of sample multi-model data
- MM-evocat [13] - a tool that allows propagation of changes from the conceptual model to the data instances

In these tools, the conceptual model of multi-model data is represented by the so-called schema category [11], which can be viewed as a multigraph. Integrating these tools provides a comprehensive solution for generating and managing multi-model data. Users can either manually create a conceptual schema using MM-cat or derive one using MM-infer, and then use MM-evocat to specify the decomposition of this schema into a selected multi-model representation. The data can subsequently be transformed automatically into the appropriate multi-model format.

3 Project Execution

This chapter provides a comprehensive overview of the project execution process. The journey of this project involved not only the implementation of code but also a significant amount of additional work, which will be described in detail.

3.1 Understanding the Tools

At the start of the project, it was crucial to first familiarize myself with the tools that would be essential throughout the development process. This initial phase was not straightforward; it required a deep dive into the theory and concepts underlying these tools. To effectively utilize them, I also needed to study various academic papers [18, 12, 11, 14, 13, 15] related to their functionalities and applications. The learning curve was steep, demanding a significant amount of effort.

The first step also included the deployment and status analysis of the three existing tools—MM-cat, MM-evocat, and MM-infer. This involved examining differences from the latest version of the schema category, variations from the most recent set of editing operations, and the currently supported models, formats, and their versions or limitations [13].

As the project progressed, I quickly realized that a successful execution also required the ability to learn and integrate new technologies and frameworks into the workflow. Throughout the project, I acquired proficiency in various new tools and frameworks, such as different database management systems, the Git [5] version control system, Docker [3] for containerization, Developer Tools [6] for debugging and optimization, Adminer [21] for database management, and server management skills.

3.2 Integration

The integration phase of the project involved combining two essential tools, MM-cat and MM-infer, into a cohesive system. Its technical details are described in the Programmer's documentation as well as in code comments. Here I outline some of the key design choices.

3.2.1 Integration of MM-infer into MM-cat

The integration of MM-infer into MM-cat was approached with the goal of preserving the modular structure of MM-cat while enabling seamless interaction between the tools. To achieve this, MM-infer was included as a separate module within MM-cat. This design

choice allowed MM-infer to function as a library that could be independently updated or extended without affecting the rest of the system.

3.2.2 Conversion Component

However, since MM-cat and MM-infer operated on different schema representations, a critical challenge was ensuring compatibility between their outputs and inputs. MM-infer produced schema representations in the form of Record Schema Descriptions (RSD) [14], while MM-cat used a Schema Category based on category theory for modeling multi-model data. To bridge this gap, I designed and implemented a Schema Conversion component. This component is responsible for transforming the output of MM-infer (RSD) into the input format required by MM-cat (Schema Category).

3.3 Other implementation work

In addition to developing the conversion module, the project required a significant amount of other implementation work. This chapter outlines the key areas of additional development work.

3.3.1 Creation of new Wrappers

One of the extra implementation tasks was the creation of new wrappers for handling JSON and CSV data formats. This included the development of wrappers for the inference process as well as the processes of Instance Category creation and final data generation. The outcome was the extension of the capabilities of the tool enabling it to accept a broader range of data input.

3.3.2 Unification of Data Sources

Originally, MM-cat was designed to work separately with databases and data sources. To streamline its functionality and reduce redundancy, we unified these two under a single concept of "Data Sources." This change required modifying the existing architecture and refactoring the codebase to ensure all references to databases and data sources were updated accordingly. The unification simplified the system's operation, making it more intuitive and easier to manage within the integrated tool.

3.3.3 Integration and Adjustment of the Candidate Miner Package

A new candidate miner package was added to enhance the integrated tool's capabilities. However, since this package was not fully finished when initially incorporated, it required adjustments and modifications to function correctly within the integrated environment. This involved adapting the existing code, fixing incomplete functionalities, and ensuring that the package was compatible with the other components of the integrated tool.

3.3.4 Development of the Final Pipeline

Creating the final pipeline was a critical part of the implementation process. This pipeline needed to handle the full flow of operations, from running inference on the server to communicating results back to the client. I developed the logic to manage this process, establishing the necessary communication between the server and client. Fig. 1 shows the architecture of the final pipeline.

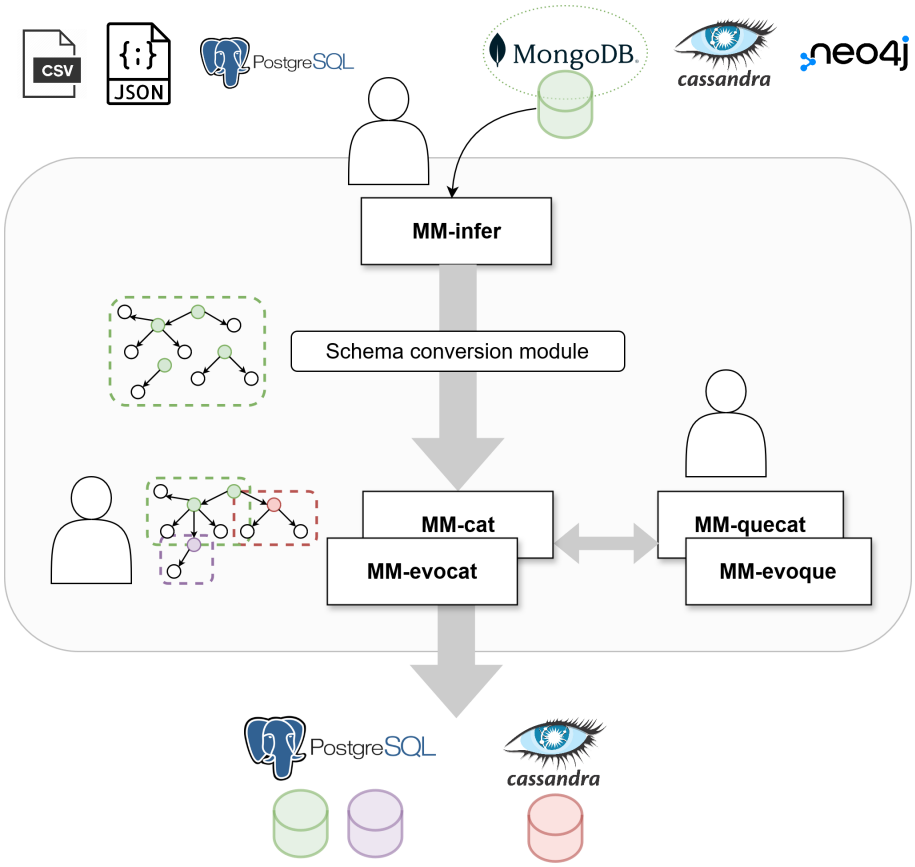

Figure 1: Architecture of the pipeline [9]

Table 1: Edits available in the inferred Schema Category Editor

Edit	Function	Purpose
Primary Key Merge	Connect Primary Key objects with objects identified by them.	Create coherent Schema Category. Reduce redundant Primary Key identification
Reference Merge	Connect Reference and Referred Object in the schema	Create coherent Schema Category.
Cluster Merge	Replace multiple cluster members with a cluster representant.	Decrease the visual complexity of the Schema Category.
Recursion Merge	Replace recursive structures with a simpler structure representing it.	Decrease the visual complexity of the Schema Category.

3.3.5 Improving user experience

In addition to the technical integration and development work, a significant focus of this project was on improving the user experience with the integrated tool. This chapter outlines several key enhancements made to optimize the usability and overall experience for the user.

Editor for Inferred Schema Category Once the final pipeline has been finished. The integrated tool was tested with several real-world datasets. By doing this, we realized the importance of providing users with a way to organize and refine the inferred Schema Category. The datasets were of various kinds and the inferred Schema Categories reflected this variation. They also revealed some redundancies and discrepancies in the data which translated to the schema and made the schema less readable. To enhance the user experience, we developed an editor specifically for this purpose. This editor allows users to apply a set of edit functions directly to the Schema Category. The real-time editing capabilities enable users to immediately see the effects of their changes, giving them the flexibility to keep only those edits that they find beneficial. Table 1 lists the edits available and their purpose. And figures 2 and 3 depict an example of edits on an inferred Schema Category. The edits include merging by primary key and clustering.

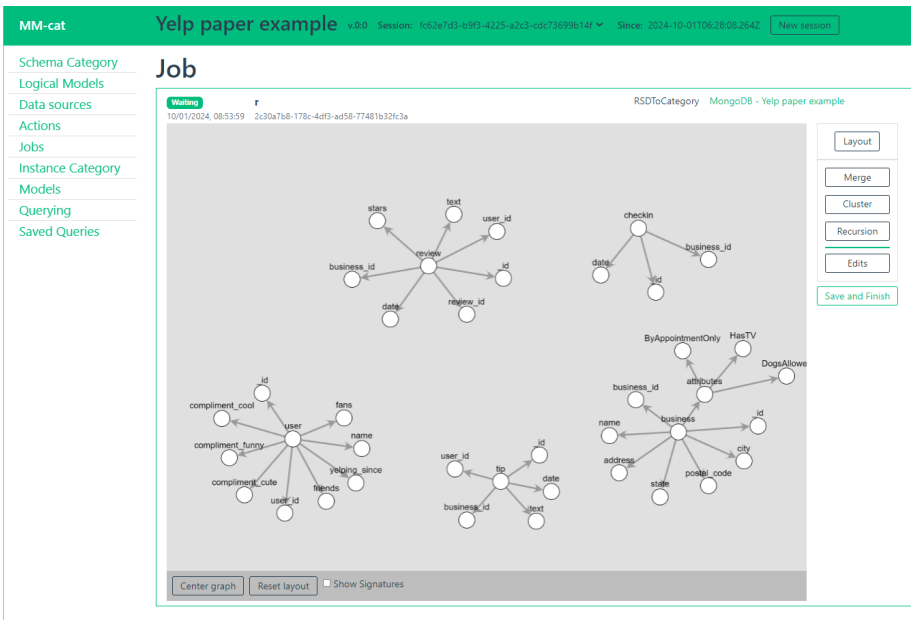

Figure 2: Editor for Inferred Schema Category with the unedited inferred schema of the Yelp dataset

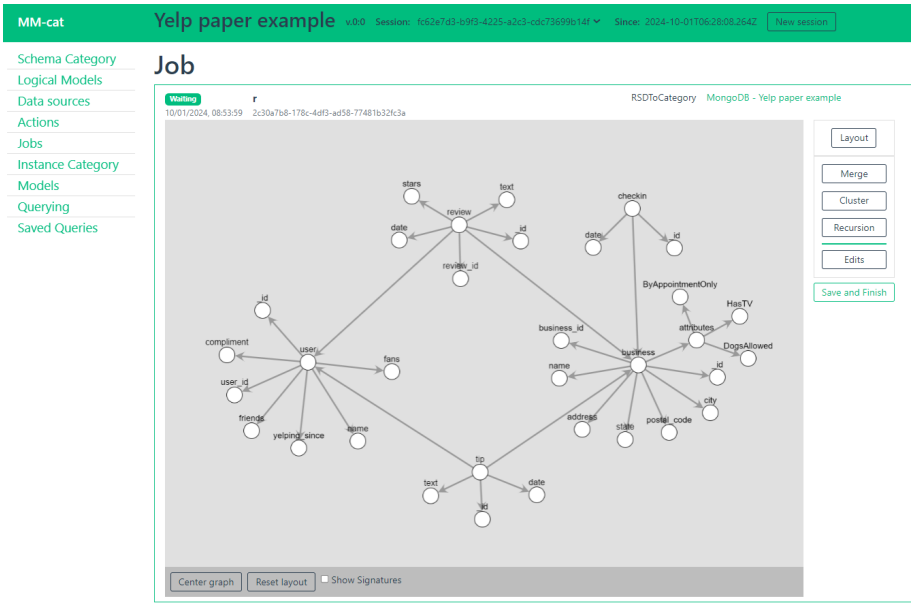

Figure 3: Editor for Inferred Schema Category with the edited inferred schema of the Yelp dataset

Candidates Building upon the editor's functionality, we further improved the user experience by introducing a feature that suggests candidate edits. Users can either create their own edit operations or select from a list of suggested edits generated by the tool. These candidate edits provide a helpful starting point for users, streamlining the editing process and enabling them to make more informed decisions about the structure and organization of the Schema Category. Fig. 4 shows an example of inferred reference merge candidates suggested by the tool for a concrete dataset.

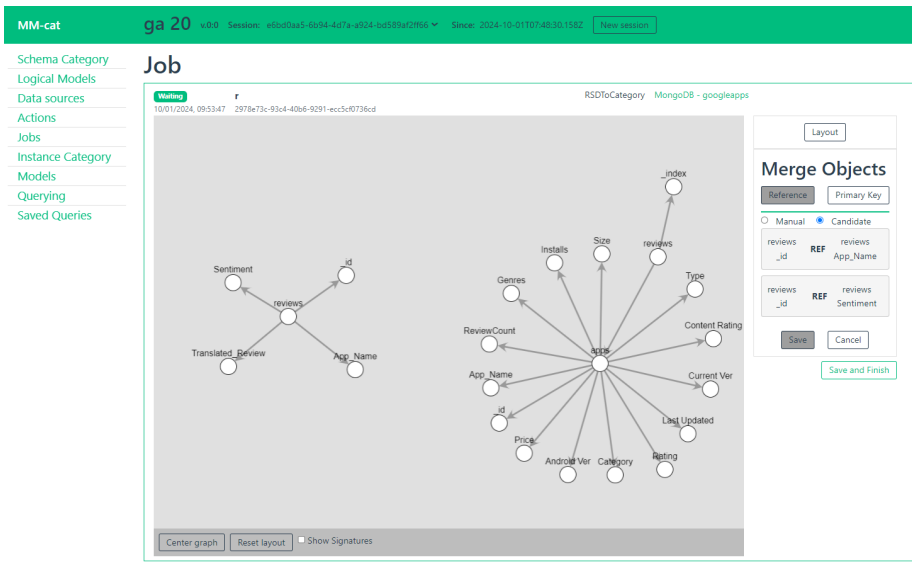

Figure 4: Editor for Inferred Schema Category with the inferred schema of the Google Apps dataset. Showing the Reference Merge Candidates.

Layout The MM-cat tool for Schema category display and editing, originally used a very simple layout to display schemas. After the exploration of several schemas inferred from complex datasets this approach was no longer applicable. The simple layout failed to meaningfully display complex structures from these datasets. Recognizing that graph layout is a complex challenge and that a single layout may not suit all datasets, we added multiple new layout options for displaying the Schema Category. Users can choose from various visualization options, allowing them to select the layout that best fits their specific dataset. The user is also able to manually adjust individual nodes of the schema and then save the new positions.Fig. 5 through 8 show the layout options newly available in the tool.

Mapping Editor To further enhance the user experience, we developed a more advanced mapping editor. This new editor streamlines the process of creating mappings by offering a context menu and keyboard shortcuts, reducing the need for repetitive manual actions. These new features enable users to work more efficiently, speeding up the mapping creation process and making it more intuitive. The current editor version supports the creation of new kinds, addition and deletion of objects and the ability to set the mapping root. However may the user want to work with nodes individually the editor also still provides the original logic for adding and editing individual properties.

Hierarchical Task Analysis Furthermore, I also performed a Hierarchical Task Analysis (HTA) (fig. 9) of multi-model data generation to reveal the steps and interactions

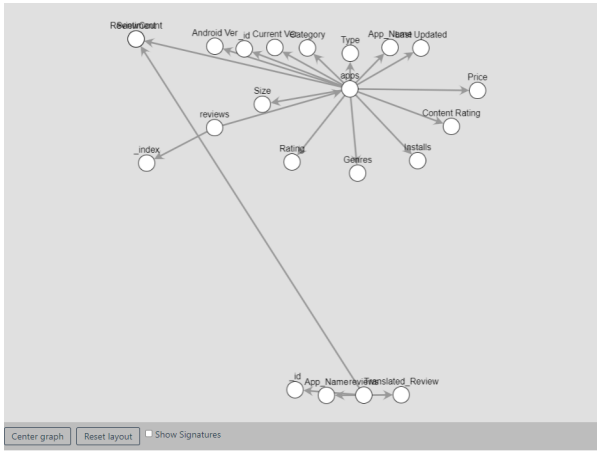

Figure 5: Force-directed Layout

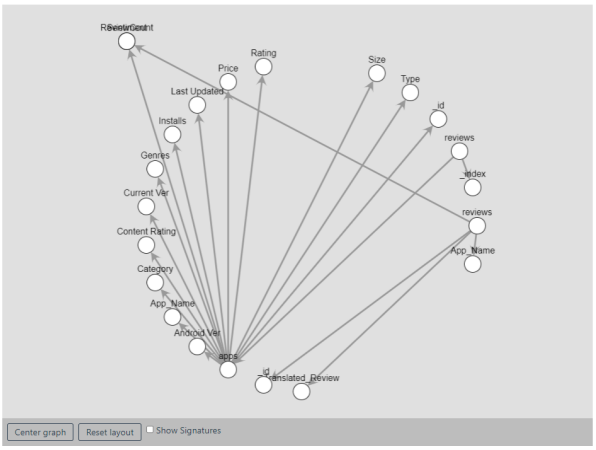

Figure 6: Circle Layout

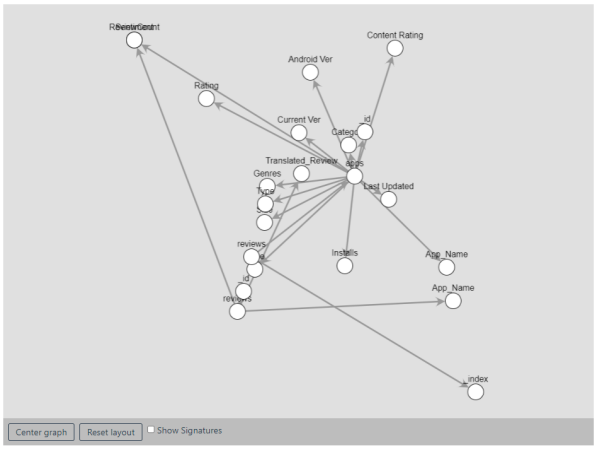

Figure 7: Kamada-Kawai Layout

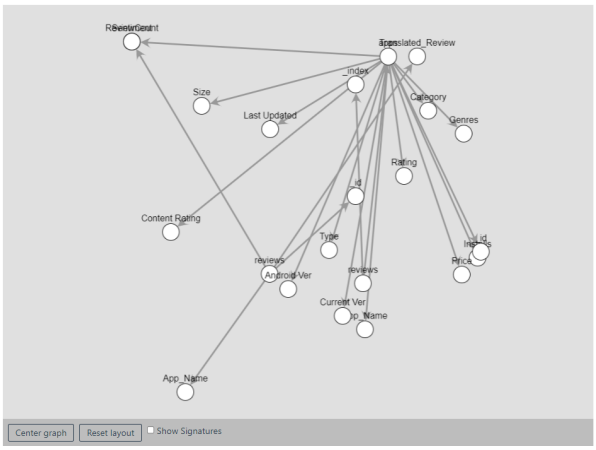

Figure 8: ISOM Layout

users will have with the tool while performing this task. The analysis helped highlight key areas for design improvement and provided insights into optimizing the flow for end users. By it, we can distinguish how the pipeline's design and usability can be further refined in the future.

3.3.6 Constant Adaptation to Ongoing Code Development

Another coding effort included the continuous adaptation of my code to accommodate ongoing updates made by colleagues to other parts of the project. This required staying actively engaged with the evolving codebase, understanding the changes being implemented by others, and modifying my own code accordingly to ensure compatibility and integration. This process demanded a high degree of collaboration, communication, and flexibility. The constant adaptation was vital to ensure that the integrated tool remained functional and up-to-date with the latest developments.

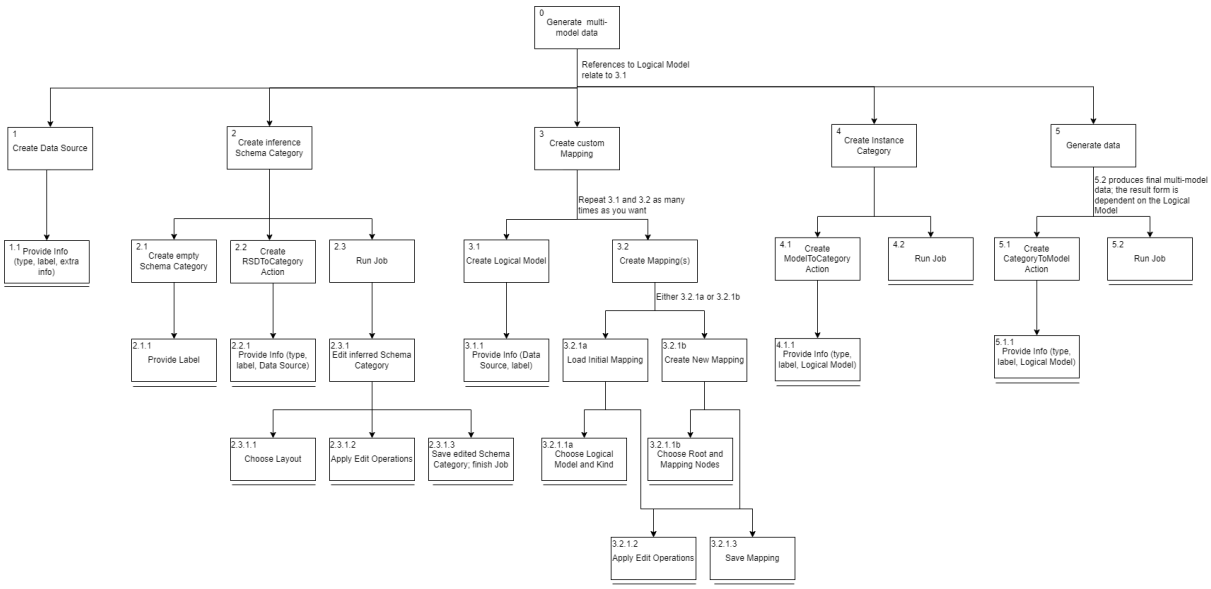

Figure 9: Hierarchical Task Analysis of multi-model data generation

3.4 Testing and Validation

To ensure the reliability of the integration and verify the correct operation of the critical parts of the inference process, I developed and executed several integration tests. These include particularly tests for the correctness of schema conversions during the inference process. Multiple tests cover different stages of the process at different conditions. Furthermore, tests for the newly implemented wrappers and editor are also included.

An important part of testing also involved exploring diverse real-world datasets, which gave us insight into the practical applications of our tool. By examining different data structures and real-world scenarios, we identified common data patterns and specific requirements that users might encounter. We gained understanding of the kinds of transformations, mappings, and features that would enhance user experience. These insights influenced our decisions on feature development and refinement. More on data exploration in the next chapter.

3.5 Dataset Exploration

The project being not only software project also included a large amount of Data Science work. The search for suitable datasets was an ongoing process throughout the course of this project. This chapter outlines the process undertaken to locate, evaluate, and select the datasets that best suited the goal of multi-model data generation.

The journey to find the right datasets involved searching though numerous repositories [10, 19, 1, 20, 8, 7, 4], each offering a multitude of datasets with varying characteristics

and structures. The primary repositories explored included: our own server, Kaggle, ... (also different types like network, graph...). Each of these repositories contained large numbers of datasets. However,the datasets necessary needed to possess a certain level of complexity and have interrelated data points to provide the foundation necessary for generating diverse models.

3.5.1 Criteria for Dataset Selection

- **Size**: As in every data science project the size of the data matters. In our scenario, datasets which would be too small would generate small amounts of multi-model data, which would appear as uninteresting for the project. Datasets with endless amounts of data would be overwhelming and incomprehensible for the viewer. Therefore the choice was intermediately sized sets of data.
- **Complexity**: The datasets needed to have a complex structure, involving multiple entities and relationships. This complexity was crucial to ensure that the data could support a variety of models.
- **Interconnections**: Another requirement was the presence of meaningful relationships among the data points.
- **Data Quality and Completeness**: High-quality datasets with minimal missing values and comprehensive data points were favored to reduce the amount of preprocessing required and ensure the reliability of the generated models.
- **Diversity in Data Types**: For the purposes of this project, datasets with a variety of data types (e.g., numerical, categorical, text, and temporal data) were considered more valuable, since on such datasets we could display the versatility of the tool.

The dataset which would posses all of these qualities would be an excellent candidate for the process of multi model data generation. Although we did not expect and also did not find a dataset which would posses all of them, we have found a few which possessed some and had properties good enough for the purpose required.

After setting the requirements the process of evaluating datasets could begin. It was labor-intensive and involved a detailed examination of each potential candidate. It could be divided into two stages:

1. **Initial Screening**: Datasets were first reviewed based on their metadata and summary descriptions provided in the repositories. Datasets that did not meet the basic criteria were quickly discarded.

2. **In-depth Analysis**: For datasets that passed the initial screening, a more in-depth analysis was conducted. This involved downloading the datasets, performing exploratory data analysis (EDA), and assessing their suitability for the project's specific needs.

Selected Datasets and Their Characteristics [Describe datasets you chose and why.Also explain which datasets you chose not to use and why.]

The chosen and processed datasets were then run through the pipeline. Before multi-model data could be generated we needed to come up with meaningful mappings to different data models for the chosen datasets. This process is straight forward for the common user, because they might have a clear idea of which data models they want to use. However for the demonstrational purposes of my project I had to come up with my own mappings. It required an understanding of both the data and the models to ensure accurate and useful mappings.

3.5.2 Real-world Datasets

In this chapter, I will briefly describe the real-world datasets we worked with. Each dataset presented unique features and complexities, guiding our decision-making process on how best to handle, or map the data within our tool. This exploration served as a foundation for aligning the tool more closely with practical, real-world applications. A thorough description of the datasets as well as their multi-model data generation process can be found in the data archive DaRe.

Yelp The Yelp dataset includes business information, including reviews, user data, check-ins, and business attributes, offering a view of consumer interactions and feedback. It is structured in JSON format, with each file containing distinct data types, such as business.json for business details, review.json for user reviews, and user.json for user profiles. For this well-structured dataset, we developed three distinct mappings to capture various aspects of it.

IMDb The IMDb dataset contains multiple files in TSV format, covering a wide range of information about films, TV shows, and media professionals. Key files include:

- name.basics.tsv – Lists individuals involved in media (e.g., actors, directors), with birth/death years and professions.

- title.crew.tsv – Details directors and writers for each title.
- title.basics.tsv, title.akas.tsv, title.ratings.tsv – Provide core details, title aliases, and viewer ratings.

To start with, we first identified primary keys to find optimal linking between files. The best mapping involved embedding data from name.basics and title.crew, allowing connections between creators and their projects. Later, by closely examining title.crew, we chose to split this file, separating it into data for writers and directors to improve clarity and facilitate more targeted analyses.

Housing in London Boroughs The London Boroughs dataset provides detailed metrics on housing in various London boroughs. This dataset has the structure of a data warehouse export, making it suitable for demonstrating our tool's ability to infer schemas from data warehousing sources. Through it, we successfully generated fact tables and relationships, proving compatibility with data typically found in warehouse systems. Although we won't include it in further analysis, this dataset highlighted our tool's flexibility with this data type.

BibleData The BibleData dataset is a complex dataset containing structured information on Bible texts, translations, and metadata. We used only a portion of it, focusing on its relational and graph-like structures, which made it ideal for a Neo4j mapping.

NASA The NASA dataset consisted of a single large JSON document detailing NASA's various code projects. The inferred schema from this dataset differed notably from those encountered previously, reflecting the unique structure of project-based data. Due to this structure, we opted to map it to relational tables, which allowed us to organize the data more efficiently within a structured, tabular format. This approach enabled better access and manipulation of the project data.

LDBC SNB Social Network Data The LDBC SNB Data Generator [2], produces synthetic graph data. We chose the social network dataset, which mimics realistic interactions in a social network. The generated data included relationships and attributes across more than 20 CSV files, such as user profiles, posts, and interactions. This dataset was chosen due to its role as a data source for the UniBench paper [17], which underscored its utility in benchmarking multi-model and complex data processing tools. However, after

looking at the data and trying to infer its schema several issues were encountered. For instance, some files had duplicate column names, which is not valid for CSV standards. Additionally, there was inconsistency in headers: while some CSV files included them, others did not. On the other hand, the dataset's structure—where some CSV files represented entity types (kinds) and others denoted relationships—was intriguing, and our tool was capable of inferring these distinctions effectively. Nonetheless, In the light of the state of the data we decided not to proceed with it. However, the dataset still served well in discovering new types of real world data and its challenges.

SWAPI The SWAPI dataset - data from the Star Wars universe, offers a rich and interconnected structure that makes it particularly interesting for exploration. It contains various entities such as characters, planets, films, species, starships, and vehicles, all of which reference each other to create a web of relationships. Notably, many objects feature bidirectional references, where an entity not only links to another but is also referenced back in return, forming a dynamic and intricate graph. This characteristic prompted us to embed this dataset.

4 Next Steps and Dropped Concepts

The process of developing the tool was not just a technical project, but also a journey of developing new ideas. As the project progressed, several new ideas emerged, inspired by the challenges and opportunities encountered during development. Some of them quite promising, some not so much. A few concepts were tested but eventually discarded due to their lack of practicality or relevance. This chapter outlines these evolving ideas.

4.1 Ideas with Potential

Some ideas surfaced that have the potential to enhance the tool's functionality and user experience. These ideas have a high probability of being implemented in upcoming iterations:

- **Extending Input and Output Formats with New Wrappers**: This enhancement would allow the tool to support a wider range of data formats, increasing its versatility and usability. Although this project has already extended the tool with a few new wrappers, the further extension is an ongoing process.

- **Query transformations**: As proposed in [9], the tool could be extended to transform data models as well as queries over the data. This would be done by incorporating the logic implemented in [16].
- **Creating Schemas from Logical Definitions**: Instead of generating schemas directly from data, an alternative approach is to create schemas from logical definitions. This method would involve using logical schemas such as JSON Schema, XML Schema, or DDL commands in SQL to define the data structure. This approach could simplify the process of generating complex schemas and enhance the tool's flexibility.

4.2 Discarded Ideas

Through the course of the project, some ideas were found to be less beneficial or redundant for the tool's intended purpose. These concepts, while initially promising, were ultimately discarded:

- **Generating Final Schema of Output Data**. The enhancement of the output algorithms for data generation by including the generation of the final schema of the data was discarded. Although this feature was implemented, it was determined that it did not add significant value for the user and would not be widely utilized.

5 Conclusion

The development of the multi-model data generation tool has been a complex and multi-faceted project. This report has detailed the various stages of the project. To conclude, here are the may contributions of this project:

- **Introduction of inference functionality** (integration of tools)
- **Extension of input types** (addition of wrappers)
- **Addition of specialized functions** (Candidate miner package)
- **Improvement of user experience** (Inference and Mapping Editor, layout options)
- **Support for creating multi-model data from single model inputs**
- **Advancement of the team's progress and stimulation of new ideas**

References

[1] CERN. Zenodo, 2024. URL https://zenodo.org.

[2] Linked Data Benchmark Council. Linked data benchmark council social network benchmark (ldbc snb), 2024. URL https://github.com/ldbc/ldbc_snb_datagen_spark.

[3] Docker, Inc. *Docker*, 2024. URL https://www.docker.com.

[4] Open Knowledge Foundation. Datahub, 2024. URL https://datahub.io.

[5] Git contributors. *Git*, 2024. URL https://git-scm.com.

[6] Google, Inc. *Chrome Developer Tools*, 2024. URL https://developer.chrome.com/docs/devtools/.

[7] U.K. Government. data.gov.uk, 2024. URL https://data.gov.uk.

[8] U.S. Government. data.gov, 2024. URL https://www.data.gov.

[9] Irena Holubova, Alzbeta Srutkova, and Jachym Bartik. Reshaping reality: Creating multi-model data and queries from real-world inputs. The 40th ACM/SIGAPP Symposium On Applied Computing, under review, 2024. Manuscript submitted for publication.

[10] Kaggle, Inc. Kaggle, 2024. URL https://www.kaggle.com.

[11] Pavel Koupil and Irena Holubová. A unified representation and transformation of multi-model data using category theory. *Journal of Big Data*, 9(1), may 2022. doi: 10.1186/s40537-022-00613-3.

[12] Pavel Koupil, Martin Svoboda, and Irena Holubová. Mm-cat: A tool for modeling and transformation of multi-model data using category theory. In *MODELS '21: Proceedings of the 24th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems*. ACM, October 2021. ISBN 9781665424851.

[13] Pavel Koupil, Jáchym Bártík, and Irena Holubová. Mm-evocat: A tool for modelling and evolution management of multi-model data. In *Proceedings of the 31st ACM International Conference on Information amp; Knowledge Management*, CIKM '22. ACM, October 2022. doi: 10.1145/3511808.3557180.

[14] Pavel Koupil, Sebastián Hricko, and Irena Holubová. A universal approach for multi-model schema inference. *Journal of Big Data*, 9(1), aug 2022. doi: 10.1186/s40537-022-00645-9.

[15] Pavel Koupil, Sebastián Hricko, and Irena Holubová. Mm-infer: A tool for inference of multi-model schemas. In *Proceedings of the 25th International Conference on Extending Database Technology*. OpenProceedings.org, April 2022. ISBN 9783893180813. doi: 10.48786/EDBT.2022.52.

[16] Pavel Koupil, Daniel Crha, and Irena Holubova. Mm-quecat: A tool for unified querying of multi-model data, 2023.

[17] Jiaheng Lu et al. Unibench: A benchmark for multi-model database management systems. In *Proceedings of the VLDB Endowment*, 2019. URL https://www.cs.helsinki.fi/u/jilu/documents/UniBench.pdf.

[18] Martin Svoboda, Pavel Čontoš, and Irena Holubová. *Categorical Modeling of Multi-model Data: One Model to Rule Them All*, pages 190–198. Springer International Publishing, 2021. ISBN 9783030784287. doi: 10.1007/978-3-030-78428-7_15.

[19] Harvard University. Harvard dataverse, 2024. URL https://dataverse.harvard.edu.

[20] Stanford University. Stanford network analysis project (snap), 2024. URL https://snap.stanford.edu.

[21] Jakub Vrana. *Adminer: Database management in a single PHP file*, 2024. URL https://www.adminer.org.