Universiteit Leiden
The Netherlands

Opleiding Informatica

Extraction, transformation, linking and loading of cultural heritage data

Michael de Koning

Supervisors:
Prof. Dr. Ir. Wessel Kraaij
Ir. Richard van Dijk

BACHELOR THESIS

Leiden Institute of Advanced Computer Science (LIACS)
www.liacs.leidenuniv.nl

16/08/2022

Abstract

The interdisciplinary project between LIACS and the humanities faculty called Linking University, City and Diversity (LUCD) aims to visualize the interaction between the university and the city of Leiden since the founding of the university in 1575 by applying data science techniques. This thesis combined with two other Bachelor's theses lay the foundation of a software architecture which includes adapters for extracting data from multiple sources, a central database and various visualizations of the data. In this thesis, software components, i.e. adapters, are developed that perform an extraction, transformation, loading and linking process on sources containing historical entities and an enhanced database model design is proposed. There are multiple adapters to extract data from Excel files and an API. Data collection is done in a manner so that not all available data is loaded into the database, but rather to initially select a set of high quality entities and enrich those entities with data that is less reliable. This is to ensure the overall reliability of the central database. The enrichment of entities is done with record linkage. The enhanced database model is based on the work of one of the other bachelor's theses involved in the LUCD project. The enhanced database model while still being able to store all relevant data and the relation is significantly reduced in terms of complexity. This reduces the query complexity for both data retrieval and data insertion and theoretically improves the execution time.

Acknowledgements

I would like to thank Wessel Kraaij for his guidance during the writing of my bachelor's thesis and Richard van Dijk for providing his knowledge and expertise, as well as his guidance and contributions towards my project. I am also grateful to Liam van Dreumel and Rick Schreuder for our collaboration during our theses. Lastly, I would like to thank Ariadne Schmidt for providing information about the LUCD project.

Contents

1 Introduction

1.1 Background

The interdisciplinary project Linking University, City and Diversity[1] (LUCD) aims to visualize the interaction between the university and the city of Leiden since the founding of the university in 1575 by applying data science techniques. The LUCD project is a cooperation between a group researchers and students from LIACS and the humanities faculty. The "core" team consists of of Wessel Kraaij (Professor of Applied data analytics), Ariadne Schmidt (Professor by special appointment of History of Urban Culture, in particular of Leiden), Joost Visser (Professor Large Scale Software and Data Science) and Richard van Dijk (research software engineer). There currently are three Bachelor's projects, including this project, to support the LUCD project. Liam van Dreumel works on the visualization of the data, Rick Schreuder works on the design of the database and I am responsible for the software components, called adapters, for the extraction, transformation, loading and linking of the data. The Bachelor's projects are interlinked and together form the design and implementation of the software architecture developed by Richard van Dijk, see Figure 1. The ultimate goal is to develop a tool for historians which can assist them in answering specific research questions. Additionally, the project aims to support the creation of visualizations and websites for the public at large.

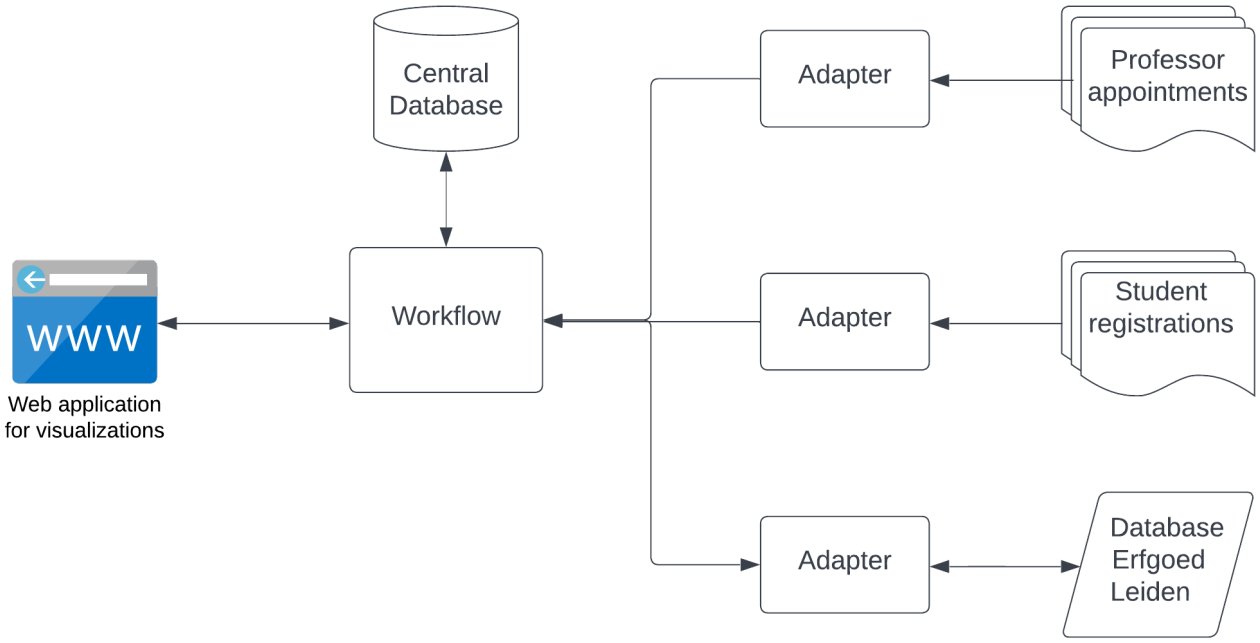

Figure 1: Software architecture of the LUCD project.
Note: only bidirectional arrows represent online data flows.

[1] https://www.universiteitleiden.nl/onderzoek/onderzoeksprojecten/wiskunde-en-natuurwetenschappen/liacs-linking-university-city-and-diversity

1.2 Objectives

The main objective for this thesis is to extract, transform, load and link data from sources containing historical entities. An entity could for example be a person or a building. Given different types of sources the relevant data must extracted. Then it must be transformed from the original format to the format of the database. The entities that will be loaded into the database must also be linked by applying record linkage. In other words, the data must not be inserted to the database just like that without going through some process where entities are linked first. Finally, the data will be loaded into the database. This should all be handled in a so called workflow. In short, the workflow regulates the requests made from the web application while also connecting the adapters to the central database. It is important the adapter components perform well in terms of efficiency, accuracy and unification capability.

The database is a given component, however it is limited in terms of completeness. As the sources contain a lot of useful data it is uncertain whether the current database model covers all data. If necessary, the database must be enhanced to encapsulate all relevant information.

1.3 Research questions

The main research question is: How can we accurately extract, transform, link and load cultural heritage data from multiple sources with a historical ontology using software components? We will answer this by looking at several subquestions:

- How can we extract, transform and load data from multiple sources?
- What strategy should we use when it comes to adding data to the database?
- How can we link different occurrences of the same entity from multiple records?
- How can we enhance the given database model to reduce its complexity in such a way that it also still encapsulates all relevant information?

1.4 Thesis overview

First we discuss relevant academic work related to this thesis in section 2. The sources used, the data collection strategy and the processing of the data are outlined in section 3. An analysis on the given central database and results of the analysis are discussed in section 4. The way the adapters handle the sources are explained in section 5. The conclusions and discussion along with the limitations of this thesis are given in section 6. Finally, further research is elaborated on in section 7.

2 Related Work

2.1 ETL process problems

A significant part of this thesis concerns the extraction, transformation and loading (ETL) process. An ETL process can be defined as the data integration process which combines multiple data sources into a single database or data warehouse. There is a plethora on previous research regarding the ETL process in data warehouses. As this thesis has no data warehouse but rather a single database, a lot of the research is not directly relevant. However, one of the major issues in ETL processes that is relevant is the quality of data. Marsh [Mar05] outlines the costs and the consequences of erroneous data and states that it costs organizations time and money. To show the impact of dirty data here are a few quotes from Marsh: "Business intelligence (BI) projects often fail due to dirty data, so it is imperative that BI-based business decisions are based on clean data" and "75 per cent of organisations have identified costs stemming from dirty data". Debbarma et al. [DND13] analysed data quality and the performance issues in an attempt to uncover the root causes of erroneous data. They also state that erroneous data negatively impacts organisations.

2.2 Record linkage

There have been made a lot of advances in linking Dutch civil records during recent years. This is mostly due to the work of CLARIAH, a Dutch organisation that "develops, facilitates and stimulates the use of Digital Humanities resources and infrastructures". They also have strong ties to CLARIN and DARIAH, both are similar organisations that operate on a Europe-wide level.

CLARIAH developed a tool called Burgerlinker which was introduced in the paper titled "Linking Dutch Civil Certificates" by Joe Raad et al. [RMR+20]. The authors outlined a way to link civil records and designed a Knowledge Graph which represents the relations between a person and the civil records of life events (i.e. birth, death and marriage). With extensive data pedigrees of persons can be created over multiple generations.

Schraagen [Sch14] explored the subject of historical record linkage in a database. Aspects from multiple scientific disciplines were incorporated of which each addresses a subset of problems regarding record linkage. Together it can contribute to solutions or a deeper understanding of record linkage.

3 Data Collection

Within the scope of this thesis, data about the professors and students of the university and the residents of Leiden since the founding of the university in 1575 is paramount. This data is distributed over multiple sources which will be extracted by software components called adapters. In this section we discuss which sources we use and what strategy we adopted to collect data.

3.1 Sources

Historical data is being digitised more and more. Digitised data however is a broad concept as there are no requirements other than that it must be available on digital devices. It could be in the form of scans of images, XML files, CSV files, Excel files, databases and many more. The digitised sources used for this thesis however are already in some machine-readable form.

3.1.1 Professors and students

The data sets containing information about the professor appointments and student registrations were provided by the university with special thanks to Martine Zoeteman (PhD History, Leiden University), Saskia van Bergen (Head Services & Collection Information Special Collections, University Library Leiden) and Stelios Paraschiakos (Data Scientist/PhD candidate). The data sets were originally stored in a Microsoft Access database but were later converted to Excel files with help of Wessel Kraaij. In both data sets the records are stored row-wise. The attributes of a person, e.g. name, birth date and expertise, are stored in the columns. The data sets of the professor appointments and student registrations do not contain the same attributes. Although there is some overlap between the two data sets, there remains a substantial difference. The most significant difference is that the professor data set contains more detailed information about their origin, life course and the engagement(s) they have, had or have had during their time at the university. The students data set only points out the abbreviated name of the faculty they studied at, no specific field of study. See Table 1 for the list of relevant attributes from the data sets.

Data set	Professor appointments	Student registrations
Attributes	Full name	Full name
	Call sign	Place and year of birth
	Date and place of birth	Date of registration
	Date and place of death	Faculty name (abbreviated)
	Dissertation	Religion
	Appointment(s)	Profession

Table 1: Attributes per Excel data set N.B sf
N.B. some attributes have been merged into one comprehensive attribute name.

3.1.2 Civil registration

Information on the residents of Leiden can be found in the civil registration, a system by which the government records and keeps track of certain vital events such as births and marriages. To

access such data we cooperated with an organisation called Erfgoed Leiden en Omstreken [Lei]. They collect, store and maintain a vast collection of historical data, both physically and digitally. Erfgoed Leiden possesses civil registration files of Leiden and the surrounding area dating back as far as the year 1450.

Erfgoed Leiden stores their data in the A2A data model [MM]. The A2A data model provides a generic data format for the exchange and access of historical personal data. Each record in the A2A model consists of four main elements, Person, Event, Relation and Source. For example, let us take a single marriage record. Person contains the names and optionally some additional information of the persons involved in the marriage. Event contains what type of event takes place, a wedding, and some additional information. Relation contains the type of relation a person which in this case could be a bride, a groom or even a witness. Source contains details about the origin of the record, i.e. what kind of record it is and where it originates from. For an example of a marriage record conform the A2A data model see Appendix C.2.

All records can be retrieved from Erfgoed Leiden through OAI-PMH harvesting. However, there is no search function and the harvesting process returns an unindexed XML file. Searching for a specific records thus needs to be done exhaustively which can be very time consuming as there are roughly 2.7 million records which contain about 7 million mentions of persons. They referred us to the API of Open Archives [Cor] which among other archives stores the data of Erfgoed Leiden. The API has a search function and returns the requested data in JSON format which works very efficiently. With the API you can easily request specific records based on a name and optionally a time period.

3.2 Data collection strategy

The sources, especially the civil registrations, contain a large amount of data. We cannot just haphazardly extract all data and insert it to the database. This has several drawbacks. It is very important to take into account that not all data is clean and correct as historical data is often subject to human error [Her19]. Data needs to be preprocessed before inserting to the database and you need to carefully consider what data to insert and what not to insert.

3.2.1 Data quality

The quality of the data is very import when it comes to tools like these. Carlo Batini et al. [BCFM09] stresses that poor data quality has a negative impact on decision making. As historians will use this tool to try and answer their research questions that rely upon this data, it is save to say that data quality is of crucial importance. It is therefore important that the database should contain high quality data. Singh et al. [SS10] defined six dimensions of data quality. Below is each dimension with a short description.

Completeness: is all requisite information available? Consider the mandatory and optional aspects, some data should be available and for some data it is acceptable to be missing.

Consistency: values should be consistent across distinct occurrences of data sets.

Validity: correctness and reasonableness of data.

Conformity: if specified formats for the data are set, the data should adhere.

Accuracy: Do data objects accurately represent the "real-world" values they are expected to model? For example, spelling errors could impact the performance of the tool.

Integrity: data should be linked if possible. The inability to link related records together may actually introduce duplicates.

Based on the the previous among other things Souibgui et al. [SAZ+19] identified causes of data quality problems within each of the three ETL processes which need to be resolved to ensure a higher data quality, see Figure 2.

		Problems	Descriptions
E	Schema	Lack of integrity constraints [27]	Rule that defines the consistency of a given data or dataset in the database (e.g., Primary key, uniqueness). **Example of uniqueness violation:** Two customers having the same SSN number customer 1= (name="John", SSN="12663") , customer 2= (name="Jane", SSN="12663").
		Poor schema design	Imperfect schema level definition [27, 16]. **Example 1:** Attributes names are not significant: "FN" stands for First Name and "Add" stands for Address **Example 2:** Source without schema: "John;Doe;jd@gmail.com;USA".
		Embedded values	Multiple values entered in one attribute [16] . **Example:** name=" John D. Tunisia Freedom 32".
	Instance	Duplicate records	Data is repeated [9]. Misspellings, different ways of writing names and even address changes over time can all lead to duplicate entries [18]. Another form of duplication is the conflicts of entities when inserting a record having the same id as an existing record [39].
		Missing values	Yang et al. have classified missing values into two types: Data in one field appears to be null or empty [39, 19] (i.e., Direct incompleteness) and missing values caused by data operations such as update (i.e., Indirect incompleteness)[39].
T	Schema	Variety of data types	Different data types between the source and the target schema.
		Naming conflicts	If we have two data sources which have two synonymous attributes (e.g., gender/sex) then the union of the aforementioned sources requires schema recognition [19, 27, 18].
	Instance	Syntax inconsistency (Structural conflicts)	The date retrieved from the source hasn't the same format as the DW's date [39]. There are a different syntactic representations of attributes whose type is the same [9]. **Example 1:** French date format (i.e., dd/mm/yyyy) is different from that of the US format (i.e., mm/dd/yyyy). **Example 2:** Gender attribute is represented differently in the two data sources, e.g., 0/1, F/M.
L		Wrong mapping of data	Linking a data source to the wrong destination results in the spread of wrong data.
		Wrong implementation of the slowly changing dimension	Problem with versioning of data after every load and update operation [19].

Figure 2: Examples of ETL data quality problems [SAZ+19, Table 1]

Souibgui et al. [SAZ+19] and Singh et al. [SS10] stated that the quality of the ETL processes greatly influence the analyses performed on the data in the database. It is therefore very important that ETL processes are designed in such a way that they detect and resolve the data quality problems as good as possible.

3.2.2 Enrichment strategy

The other aspect of high quality data is the reliability of the source. If we for example consider the archives of Erfgoed Leiden, historical data from the archives is originally handwritten and humans are known to make mistakes. Humans may mishear something in an oral interview or inaccurately

copy data between records. Although this type of data is inaccurate to some degree, it does not mean that the data should be disregarded. This leaves us with two strategies. On the one hand we could just load all available data into the database and afterwards query for links. On the other hand we could select high quality data or sources and use that as the foundation for finding links to only then load the linked data into the database. The first strategy would mean that we basically copy all sources to our own database. There is a high probability that this strategy would leave us with superfluous and possibly inaccurate information that might never be relevant for visualizations or research questions. The second strategy would prevent this from happening. We call this the enrichment strategy. We start with filling the database with relevant data of which we can say with a high certainty that it is reliable. We then start looking for other relevant data from the less reliable sources but now with the reliable data as foundation. In other words, we only insert new data of which we can conclude to have a relation with the reliable data. When we repeat this process new relations will be uncovered with every iteration. This way eventually all data in the database can be traced back to reliable data adding to the general reliability of all data.

We consider the data related to and provided by the university as reliable because it was carefully composed by trusted persons (3.1.1). This will be added to the database first upon which the enrichment strategy builds. The person entities of the professors and students will be enriched with data from external sources by using record linkage. This will be discussed in the next section.

3.3 Record linkage

As discussed in the previous section we aim to enrich high quality entities, so for this thesis the focus primarily lies on establishing familial relations of professors and students with the goal to visualize these relations. With civil registrations, which are the authentic sources of birth, death and marriage events, it is possible to reconstruct life courses and family relations [vdBvDM+21].

Information about the professors such as the date and place of birth and death is well documented. With this information we can perform targeted searches in the data sets of Erfgoed Leiden for civil registrations of life events. High quality entities are initially matched to civil registration records based on two matching conditions, name and time period. For matching records we use the same set of assumptions as CLARIAH's Burgerlinker [RMR+20]:

- Persons will not become older than 110 years of age
- Persons can marry at age 13
- Children are born to:
  - i married parents,
  - ii up to 9 months after a married father perished,
  - iii up to 5 years before the parents married IF acknowledged by the father, or
  - iv up to 10 years before the parents married IF acknowledged by the father from birth
- Women can give birth to children between age 14 and 50 years
- Men can become father at age 14, and stop reproducing after their wife turns 50

3.3.1 Method

Before entities can be linked, the corresponding records must be matched. Raad et al. [RMR+20] and Mourits et al. [MVDM20] both discuss the matching of records in the LINKS/CLARIAH project, the predecessor of Burgerlinker. Initially, the records will be matched based on the name and a logical time period. Both use a maximum Levenshtein distance to match names. For this thesis we rely on the phonetic similarity measure provided by the Open Archives API. A matched record will then be compared to the corresponding high quality entity from the database. First of all, the matched records will be compared against the assumptions from CLARIAH's Burgerlinker. If it passes the assumptions, important attributes will (again) be compared, these are the name, the logical time period, age, place of birth, residence and parents. Attributes may be missing, but as long as non of these attributes from the high quality entity and the matched records contradict each other there is a (potential) link. Because data availability may differ we propose a simple classification. If the attributes do not contradict and at least the name and logical time period are available and correct, it will be classified as a "potential link". This will always need verification from a historian. If more attributes can be matched it will be classified as a "link". The verification by historians is part of the LUCD project. The classification of the links will be uploaded to the database so that historians will have the option to verify or debunk links. If a link can be established based on the conditions all corresponding data will be loaded into the database, see section 5.2. This may include enriching/updating a high quality entity with new found data or inserting a completely new entity such as spouses or (grand)parents of high quality entity.

4 Central Database

The collected data will be stored in a central database. Rick Schreuder [Sch22] wrote his Bachelor's thesis about the design of a database model for the LUCD project which will be discussed in section 4.1. He concluded it is best to choose the relational database management system (RDBMS) MySQL which uses the InnoDB storage system.

The design of the database was done by modelling an Enhanced Entity-Relationship (EER) diagram, a high-level data model. The current database model will be discussed in section 4.2 and is based on both the sample research questions (SRQs) in the LUCD project and the data sets. The SRQs are based on the interaction between the university and the city with the subjects of interest being mobility, geographical segregation and social integration. The SRQs grouped by subject are:

- **Mobility**
  - i Where do people come from and where do end up?
  - ii Do people stay in Leiden after their study, if not where do they go?
- **Geographical segregation**
  - i Where in Leiden did the students and/or professors live?
  - ii Did academic people live amongst the other residents of Leiden?
- **Social integration**
  - i Did students/professors marry (and have children) with residents of Leiden?
  - ii After students had completed their study, did they stay and work in Leiden?

As these questions merely form a sample of all possible research questions, the model may need to be extended over time if necessary. In order for this to be possible without producing incompatibilities between versions of the data models the design has to be flexible. For now we focus on what we have. The LUCD project is interested in visualizing the interactions between the university and the residents of Leiden. The database model thus has to encapsulate all relevant entities with its attributes and relations.

4.1 Database model Rick Schreuder

Schreuder provided a model based on the information he had of the sources and the sample research questions. Although the data he had was incomplete, he designed a database model [Sch22] which in a way comprehends all the entities necessary for the project. The model, see Appendix A for the full EER diagram, is very broad and complex. The corresponding entities and attributes are displayed clearly in Table 2. The "person" entities are Student, Professor, Father, Mother and Child. It corresponds to the LUCD research questions with the goal to capture familial relations. The other entities from Table 2 are to describe the engagement with the university. Any other relevant personal information are stored as attributes.

The entities from Schreuder's model can be divided into four clusters of entities. The clusters represent and cover the university structure, the professors, the students and the family relations. Each cluster is connected to the Person entity. Schreuder claimed that all sample research questions can be answered with the Person entity combined with the clusters. The mobility SRQs can be answered with the Location entity and the university employees and students clusters because the clusters cover all student and professor data. The geographical segregation SRQs can be answered with just the Person and Location entity. Finally, the social integration SRQs can be answered using the university employees, students and family relations clusters and the Location entity because this covers all person entities and their relations including their known residences. Schreuder states that with the cluster model every possible combination of data can be retrieved within five joins for reduced complexity and better readability.

During the developing of adapters for Schreuder's model we encountered some issues. The model consists of 26 tables. The greater this number, the more this adds to the complexity of converting the source's data format to the database format. For instance, if you want to insert the data of a professor, you would have to insert into 8 to 16 different tables depending on the available information which is not ideal. The model seems to be primarily designed for data retrieval, overlooking the complexity of data insertion. Secondly, as mentioned before the model does not include all necessary attributes and relations. This was also mentioned in the limitations section of his thesis. The model is said to be limited in completeness. The emphasis was put on accurateness to provide a foundation which can be build upon to create a model that completely covers the data sets and the requirements of the LUCD project.

Entity	**Student**	**Professor**	**Father**	**Mother**	**Child**
Attributes	First name Last name Suffix Call sign Gender IsEnrolled Date of birth Date of death Religion Nationality Student number	First name Last name Suffix Call sign Gender IsEnrolled Date of birth Date of death Religion Nationality Nobel prize Appointment Discipline Start of appointment End of appointment	First name Last name Suffix Call sign Gender IsEnrolled Date of birth Date of death Religion Nationality	First name Last name Suffix Call sign Gender IsEnrolled Date of birth Date of death Religion Nationality	First name Last name Suffix Call sign Gender IsEnrolled Date of birth Date of death Religion Nationality
Entity	**University**	**Faculty**	**Institute**	**Location**	**Study**
Attributes	Name Date of creation	Name Date of creation	Name Date of creation	Country City Street name Postal code House number Addition Location date	Name Language Croho-number
Entity	**Supporting staff**	**Employee**	**Publication**	**Experiment**	**Specialization**
Attributes	Job name Field	Start of employment End of employment	Publication date Name Score	Name Date	Name

Table 2: Overview of the attributes of each entity from Schreuder's model. [Sch22, Table 3]

4.2 Enhanced database model

The goal of designing an enhanced database model was to reduce its complexity while also making sure that all relevant data can be stored in it. We approached this by doing an extensive analysis of the available data with the sample research questions, see section 4, as guideline. We also researched what structure would be optimal for our database.
Rick Schreuder [Sch22] stated in his limitations section that the data model could be optimized by implementing a star schema or a snowflake schema. Both are multidimensional schematics with the most significant difference between the two, in terms of design, being the normalization of data. The star schema is not normalized and the snowflake schema is normalized because it uses subdimension tables. Subdimension tables are a sort of lookup tables. Snowflake schematics thus have a larger number of tables than star schematics. See Appendix B for examples of a star and snowflake schema. Iqbal et al. [IMS+20] compared and analysed both schematics against each other. Ultimately it depends on the use case and the trade-offs you are willing to make when choosing between the two schematics. We will shortly discuss the the main differences between the schematics:

- **Query complexity:** queries for the snowflake schema are always more complex. Star schematics consist of fewer tables, resulting in less complex queries which make it easier to

retrieve data.

- **Disk space:** because the snowflake schema is normalized, it will have less redundant data as values are stored only once in a subdimension table. Star schematics thus require more space and memory.
- **Execution time:** on the one hand a star schema has better query performance because of the complexity difference. On the other hand however, because snowflake schema has no redundant data it makes the queries run faster. When dealing with large data sets, star schema could take more execution time than snowflake. There is no true winner in this case as it hugely depends on the data and possible optimization techniques.
- **Maintenance and flexibility:** Snowflake schema is more flexible in modifying and extending because of normalization. Altering subdimension tables does not have an effect on the entire data set. With a star schema all records could have to be updated. This also means that a snowflake schema is easier to maintain because the data/attributes are divided over multiple tables instead of all in one large table.

Based on these differences we opted to go with a snowflake schema. We foresee that our database will contain many records and that there are many attributes which can be normalized to save disk space. This can improve execution time. We also need to take into account possible expanding of the database model, for this a snowflake schema is more suited. As for the greater query complexity, we are willing to make this trade-off as the other benefits outweigh this disadvantage.

Now that we decided on a schema and, by doing an analysis, have a complete view of what information is known combined with what is required for the LUCD project we can design the database model. A snowflake schema is a multidimensional model consisting of a fact table, dimension tables and subdimension tables (Appendix B). The fact table is the main table of the schema. For this we created the Person entity as it is the most important one that connects to all other data. Based on the sources and the sample research questions we concluded that the locations of events, engagements with the university and relations between persons are the main points of interest. For each we created an entity which act as dimension tables. Each dimension table then has "type" tables, i.e. the subdimension tables, which are used to normalize certain attributes and in the process prevent redundant data. By normalizing attributes that are common between records of dimension tables, e.g. "type of person" and "type of engagement", we can optimally prevent redundant data. Subdimension/"type" tables can also easily be extended and modified because it is just a matter of updating the table instead of updating the main table which implies that all affected records need to updated too. This improves the flexibility of the model. The Person entity thus inherits the attributes from the (sub)dimension tables. We started by modelling a conceptual model with UML, see Figure 3. The final main entities along with their attributes can be found in Table 3.

Person

The Person entity essentially replaces the separate tables for professor, student or another type of person. Person has an attribute called TypeOfPerson of type integer which is a foreign key referring

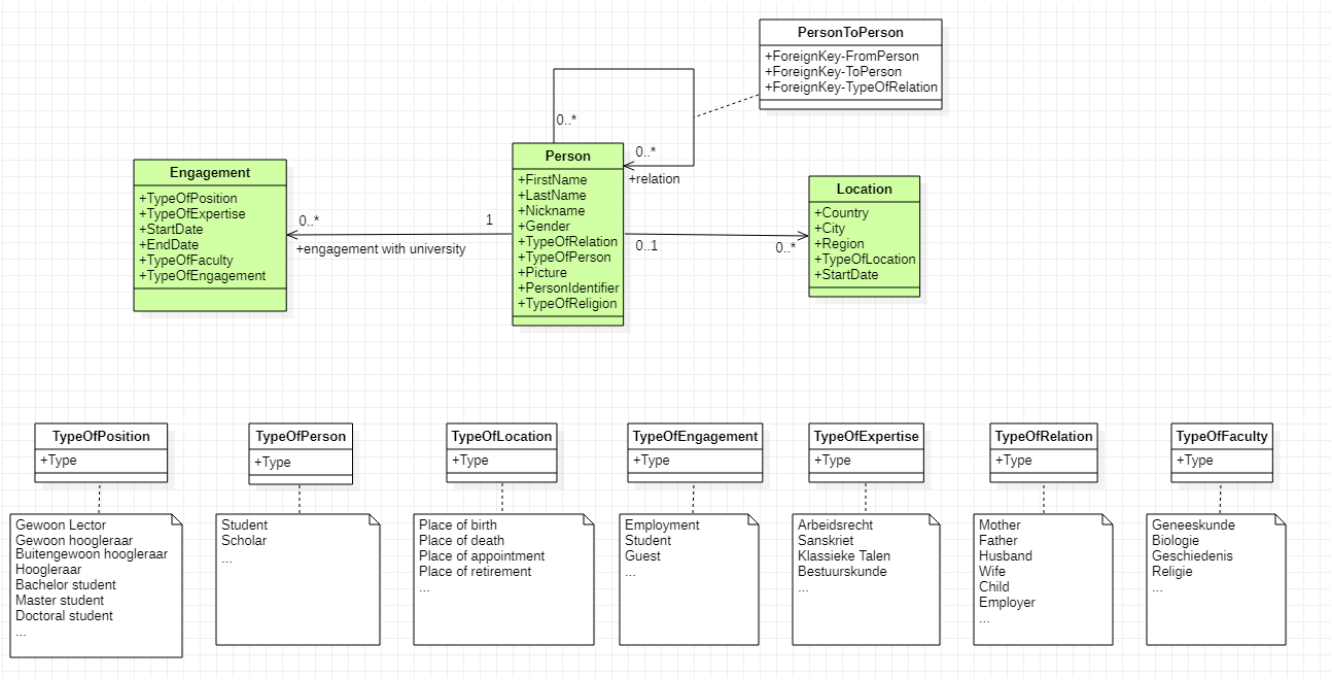

Figure 3: UML conceptual model of enhanced database model prototype

Table	Person	Location	Engagement	Relation
Attributes	Type of person First name Last name Maiden name Affix Nickname Gender Nationality Religion Status Job Source name Source rating	Type of location Country City Street House number Region Start date End date Source name Source rating Person ID	Type of engagement Type of position Type of expertise Type of faculty Start date End date Source name Source rating	Type of relation From person ID To person ID Event details Source name Source weblink Source rating Link classification Remark

Table 3: Primary tables with attributes from enhanced model

to the "type_of_person" table to resolve the issue of needing to have multiple entities. This is one of seven "type" tables or subdimension tables. It has an integer as primary key and another attribute which is a textual description of a certain "type", e.g. professor or student. This way we eliminated the use for having a table for each type of person.

Location

There are several types of locations. For example, place of birth, place of death and residence. Each person may have not one of these locations or one or more of these locations. For that reason we added a Location table which refers to Person primary key as a foreign key. That way there is a One-To-Many relationship. The type of location is also stored in a separate type table.

Engagement

To distinguish persons who are somehow directly connected to the university and those who are not we created the Engagement table. There are different types of engagement with the university, for example employment, student or guest. Those who are directly connected to the university always have one or more engagements. Professors may for instance have had multiple positions, they first could have been an assistant professor and at a later period of time be promoted to a full professor. Each engagement thus has a start and end date. Next to the type of position, the types of faculty and expertise are also attributes which are stored in subdimension tables.

Relation

To include relations between entities we created a Relation table. A person may have multiple relations, one can be a spouse, brother, son and father at the same time. Therefore Relation has a "FromPersonID", a "ToPersonID" and a "TypeOfRelation" attribute. Given a type of a relation, e.g. brother, "FromPersonID" refers to the person who is the brother and "ToPersonID" refers to the person from who "FromPersonID" is a brother. In other words, "FromPersonID" is "ToPersonID"'s brother. Both id's refer to the primary key from the Person table so all relations can be easily traced. Relation also has an attribute called link classification. This attribute contains the classification of the link made, i.e. a relation. A link can be classified as a "link", meaning that there is a very high certainty that the link is correct. It can also be classified as a "potential link", meaning that there are reasons to believe that there exists a link but it is not substantial. This requires human verification.

The enhanced database model successfully reduced the overall complexity in terms of data retrieval and data insertion in comparison with Schreuder's model. The model does differ from our original conceptual model, however this is only due to additions made during the design phase. The enhanced model satisfies all definitions of our UML diagram. For the full EER diagram of the enhanced database model see Figure 4.

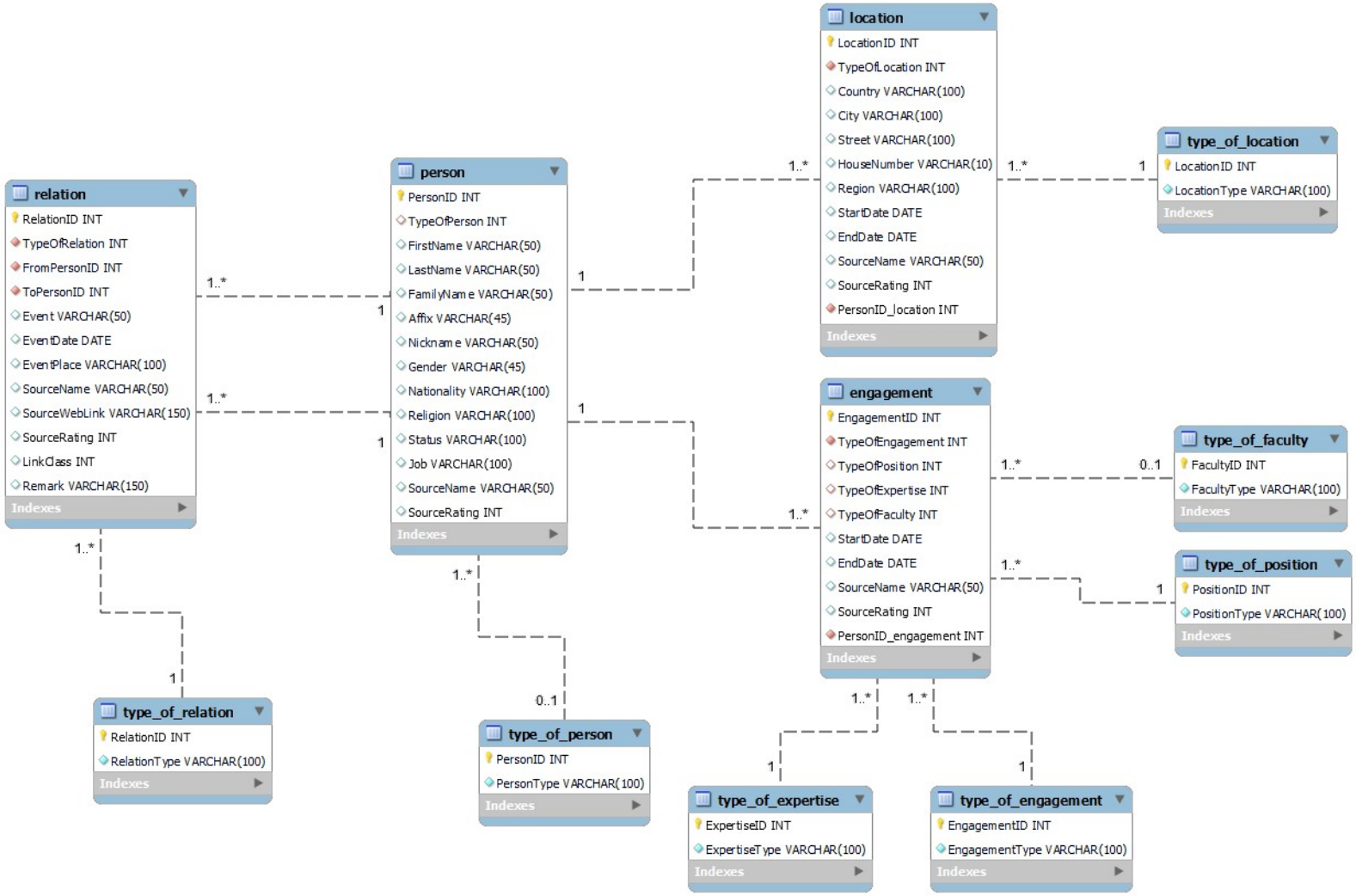

Figure 4: The EER diagram of the enhanced database model.

5 Adapters

Adapters are software components that have the function of the extraction, transformation, linking and loading of data from different types of sources. In this thesis, Python is used for all adapters. All sources used for this project have a different schema design, so each source requires a different customized adapter. See Appendix D for more information on where to find the code.

5.1 University data

The professor and student data are available in Excel files, see section 3.1.1. These data sets are labeled as high quality, so this data is inserted first to the database. First the data will be cleaned and normalized a bit based on the scripts Stelios Paraschiakos wrote. Next, the Excel files are read into a pandas[2] DataFrame[3]. The columns that contain the data corresponding to the "type" tables are inserted first to the database. The data will then be inserted per table starting with Person and followed by Location and Engagement. Per table all the relevant columns from the Excel file are selected and taken separately in another DataFrame. The data is converted to the format of the database and some verification checks are performed. Missing values will have the value "None" in the database. The DataFrame is then passed on to the database layer and inserted into the database.

5.2 Open Archives API

The adapter for the Open Archives API indirectly extracts information from the Erfgoed Leiden database and is very different compared to the adapters for the Excel files. The data from less reliable sources will not be inserted into the database without first checking whether a link can be made according to the enrichment strategy as described in section 3.2.2 and the record linkage methods in section 3.3.

The Open Archives API allows you to send parameterised queries. The query can be tweaked to limit your search results. You can for instance specify the archives you want to search and the type of source (i.e. birth, death or marriage certificate sources). This allows us to specifically search for civil registrations originating from Erfgoed Leiden. If we then also add the name and a year or period of time to the query the API returns the matching records we are looking for. The API supports phonetic search as well to also match names that sound roughly the same but are written differently. Below is an example of a request which searches for occurrences of the name "Boerhaave" based on phonetic similarity (∼) (name=∼Boerhaave) in the parish marriage records (sourcetype=BS Huwelijk) from the archives of Erfgoed Leiden (archive=elo).

https://api.openarch.nl/1.0/records/search.json?name=~Boerhaave&archive=elo&number_show=100&sourcetype=DTB%20Trouwen&start=0

The response only shows a summary of the matching records. For each matching record another request must be made with the identifiers that can be extracted from the previous request. Below the second request, with the identifier from the previous query.

[2] https://pandas.pydata.org, version 1.4.2

[3] https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.html

https://api.openarch.nl/1.0/records/show.json?archive=elo&identifier=7134512d-1ae2-94ca-cb31-6bd0a7fea6d5

See Appendix C for the responses of both queries.

Initially, for each professor and student a parameterised request will be send to the API. This request will return a response with the matching records according to the parameters. However, as mentioned earlier the response only contains part of the records. An identifier needs to be extracted from each record so that the full record can be fetched with another request. The response is in JSON format and structured according to the A2A data model, see section 3.1.2. The adapter then needs to parse the response to extract all valuable data. After the responses are converted to a DataFrame according to the database format, the adapter will then decide whether the record is an actual link according to the methods discussed in section 3.3. Like for the Excel adapters, the results will be passed on to the database layer and inserted into the database.

5.3 Workflow

The workflow can be seen as the bridge between the adapters, central database and web application, see Figure 1. A component of the workflow is the database layer, a separate layer which is the only component that can make a direct connection with the database. All functions that use a database connection are grouped and segregated from all other functions in a separate file/class. This is to ensure that the connection with the database is easier to maintain. Faults can be fixed quicker because the origin is easier to trace and updates can also be pushed through faster without to many troubles because all functions are in a separate file. For the functions of the adapters, and all other remaining functions, to make a connection with the database they have to create an instance of the "Connection" class from the database layer file.

The workflow also handles requests from the web application. Based on queries send from the web application it will retrieve and return the required data from the central database and if needed communicate with the adapters to produce a web page. However,this could not be done for this thesis and will be elaborated on later in the limitations section 6.2.

6 Discussion and Conclusions

In this bachelor's thesis, an extraction, transformation, loading and linking process for multiple sources and a database model have been developed. We aimed to answer the following research questions: How can we extract, transform and load data from multiple sources? What strategy should we use when it comes to adding data to the database? How can we link different occurrences of the same entity from multiple records? How can we enhance the given database model to reduce its complexity in such a way that it also still encapsulates all relevant information?

We extracted, transformed and loaded the data with multiple software components called adapters. By analysing the available data from the sources and comparing that with the sample research questions we knew what data we had to extract. Data quality plays an important part in the ETL process. Marsh [Mar05] states that in the BI context "corporate competence and trust are under intense scrutiny by critical stakeholders, shareholders, customers and regulatory bodies". On which he says the following, "If the accuracy and security of a company's data are in doubt, stakeholders will lose confidence, trading will be affected and performance will suffer". One could say the same in the context of the LUCD project. If our data and visualizations turn out to be inaccurate it can have consequences for the end users. If researchers cannot trust the data behind the visualizations then they will not use it to help answer their research questions. Singh et al. [SS10] outlined the possible causes of data quality problems in each step of the ETL process and defined six dimensions of data quality, see section 3.2.1. As cultural heritage data are possibly "dirty", inaccurate and depending on the source in a different format it is important to apply some methods to clean the data before actually using it for analyses or visualizations. When extracting data from the sources, the adapters make the necessary changes to the data so that the six dimensions of data quality are enforced. The following describes how the adapters enforce the six dimensions. Completeness, the data should be as complete as possible. However, the adapters extract all available data but do allow for missing data such as attributes because that is very common in genealogical sources. Consistency, there are no distinct occurrences of data sets but the adapters do extract the information exactly as it is stored in the sources. Validity, the adapters do perform validity checks, for example the age of a person is validated by comparing the date of birth and date of death to check if the age is not over the maximum of 113 years. Conformity, the data is converted to the format of the database so that all data is in the same format. Accuracy, names of for instance countries, cities and persons preferably are standardized. This is something for the adapters to improve on in further research as there is room for for more standardization. Integrity, this is an important part of the project. The adapters try to link occurrences of the same entity from different records. Another thing to enforce integrity is to update records in the database when new information is found instead of creating another instance of the same record to avoid duplicates. By enforcing the six dimensions the data is cleaned before inserting it into the database. Inserting data into the database is done through the workflow, the separate layer that connects with the database. The extraction, transformation and loading is mostly done with the use of pandas DataFrames. To ensure the adapters work efficiently we include as many build-in pandas functions which have been optimised for the use with these DataFrames. From some small tests, in which we compared the pandas functions against the use of loops and vectors, it turned out that the pandas functions have a faster execution time than the other options.

We consider two philosophies regarding the data collection strategy. Firstly, we can collect all

available data and query for matching records afterwards. Secondly, we can start of by selecting high quality data and only collect other data which enriches that data. After a consultation meeting with cooperation partner Erfgoed Leiden, the second option was preferred. In this way, data copying is restricted to data relevant to the focus of the project. The data from civil registrations is also inaccurate to a certain degree because many records are originally handwritten. We would need to process the data first to ensure the data quality. It therefore is unnecessary to store all available data in our database because there is many data not relevant to the project. Our data collection strategy is instead based on the enrichment of high quality entities. This way only new data is inserted into the database that has a (potential) link to the high quality entities. Thus only data relevant to the visualizations is stored in the database.

As of now, we extract the data from different sources and store what is relevant in our own central database including the link of the original source and when the data was accessed. Another option would be to only store a persistent link of the data which can be accessed online instead of storing it in our database. This however would require the adapters to externally access and extract all information for each entity every time that data is requested. Per request, the adapters would need to perform more operations compared to accessing the data locally in our central database which is less time efficient. This issue could potentially be resolved by implementing the principles of Linked Open Data or Linked Data. Linked Open Data[4] was introduced by Tim Berners-Lee and it is the idea of the semantic web which consists of structured, machine-readable data that is interlinked with other data on the web. All data can be shared and be read automatically by computers. This ultimately enriches the data and in turn could lead to much more useful information. Implementing the Linked Data principle in the LUCD project could have many benefits. For example, if other genealogical frameworks as well as the LUCD project follow the Linked Data principle, then our data could be linked to their data without an extensive ETL process. You would essentially get one large database of all genealogical data on the web. All data combined could then be used to uncover more information. Furthermore, because the data from multiple sources is stored in the same structure, it is easier for the computer to process the data. This would resolve or at least improve on the issue mentioned earlier about the adapters needing to perform more operations to access the data externally per request. The LUCD project could benefit from implementing the principles of Linked Data. However, this would also require the sources or other data sets to follow the same principles. This is not always the case, but there does seem to be a transition of organizations adapting the principles. Although Linked Data is not yet generally implemented, it would be a more future proof approach to do it anyway.

Record linkage is done based on the research by Raad et al. [RMR+20] and Mourits et al. [MVDM20]. Record matching through querying the Open Archives API is the first step. Subsequently a comparison between the available data from the matched records against the high quality entities will be carried out, assuming the records pass the assumptions from CLARIAH's Burgerlinker. For now, the Open Archives adapter retrieves all entries of professors from the central database. For each professor the adapter searches through the API for name matches. After parsing the API responses, the time period and other attributes from the professor are compared against the data in the matching record. The data will be compared on similarities and non-contradictory which results in a classification. The classification can be either no link, a potential link or a link. Potential links

[4] https://www.w3.org/standards/semanticweb/data

and links will be inserted into the database, including the classification. This allows historians to be able to verify or disapprove a link as to include a human second opinion. There still remains the challenge of merging multiple records of the same event. Each record of the same event may contain more or less (un)structured information than the other records of that event. The same information, such as a job description, can be written down in multiple ways. Meaning that even though two descriptions essentially say the same thing, the string is not equal. This posses the challenge to identify these equalities so that the information can be compared for similarities.

The database model designed by Schreuder consists of 26 tables. Because of the many relations and dependencies between the tables the queries can become quite complex for both data retrieval and data insertion. As proposed by Schreuder [Sch22], we optimized the model by investigating star and snowflake schematics [IMS+20]. The implementation of either schema would improve the complexity of Schreuder's model. The snowflake schema was selected because of its normalization property which makes it more flexible and prevents a lot of redundant data compared to the star schema, see section 4.2 for more details. Based on the given database model, the available sources and the sample LUCD research questions we completely redesigned the model. The enhanced database model defines the same relations as the given model but now consists of only 11 tables which greatly reduces the complexity of the queries for both retrieval and insertion. The snowflake schema also allows for easy expansion as future research questions possibly require an extended database model.

6.1 Interdisciplinary Context

The design of the adapters and the database model were guided by the input from members of the LUCD project team, our contacts from Erfgoed Leiden and the students from the other bachelor's theses.

The design requirements of the database model were primarily extracted from the input of Ariadne Schmidt (Professor by special appointment of History of Urban Culture, in particular of Leiden), who provided important information about the goal of the LUCD project. In terms of the interaction between the university and the city she pointed out the relevant subjects, mobility, geographical segregation and social integration along with corresponding research questions, section 4. From this could be deduced what information and relations should be covered by the database model. The model covers information about individual persons, relations between persons, engagements between persons and the university and locations of any type, i.e. place of birth and residence, section 4.2. We believe that with this model, visualizations can be created which can help historians in answering their research questions regarding the mobility, geographical segregation and social integration. The model in its current form could not have been realised without the input from the humanities discipline. Schmidt provided clear goals and research questions as well as providing intermediate feedback to the project which helped to shape the current model. Without this, it would have taken more time to acquire this valuable information and a possibly different and less complete model would have been created instead. For the full EER diagram, see Figure 4.

The adapters are designed to work according to the enrichment strategy, section 3.2.2. Input from our contacts at Erfgoed Leiden helped us to come up with the idea of this strategy. The adapters

thus insert high quality data first and use this as a foundation to collect more data from different sources.

6.2 Limitations

The workflow is not yet capable of processing queries send from the web application. We did not have the time to set up a local version of the server to test the implementation of the web application and the queries. This will be done in further research.

The database model can be normalized more and some other options concerning the attributes and subdimension tables need to be explored more in further research.

The Open Archives API adapter only matches marriage records and links entities for the professors. Due to a lack of time we could not link both the professor and student data sets. We chose to firstly link the professor data set because it contains more information than the student data set which makes it easier to match records and establish links. Erfgoed Leiden has two types of marriage registers, registers from the baptism, wedding and funeral (BWF) books and registers from the civil registration. An initial search shows that the names of at least 144 professors occur in the BWF books and that the names of at least 302 professors occur in the civil registration. NB There may be some overlap between the numbers.

6.3 Conclusions

Adapters and an enhanced database model were developed in order to answer the main research question: How can we accurately extract, transform, link and load cultural heritage data from multiple sources with a historical ontology using software components?

Schreuder's database model has been changed with the main goal of reducing its complexity. By analysing all available sources and combining that information with the sample research questions we were able to establish the requirements of the model. Furthermore, by investigating the implementation of an optimized structure design we were able to reduce its complexity.

The adapters perform the extraction, transformation, linking and loading of the data. The extraction, transformation and linking processes were developed according to the sources. After analysing the sources we determined what data should be extracted, how the format of the source has to be transformed and what data could be used to link data. The loading process was developed according to the structure of the enhanced database which allows for the adapters to perform less complex queries which as a result improves the performance.

7 Further Research

In further research, the methods for record linkage could be improved. It is now based on a relatively simple set of constraints but could be improved by applying machine learning techniques. There are many more sources that contain (Dutch) genealogical records. Further research could include the development of more adapters for new sources. One possibility would be to do NLP research on extracting data from non machine readable sources such as scans of newspapers. NLP research could also be applied to convert research questions into queries for the database and adapters. This could improve the convenience for the end users, i.e. the historians, of the tool.

In further stages of the LUCD project, the integration of the genealogical frameworks from CLARIAH and its international, Europe-wide counterparts could be explored. By doing this, more complete life courses of professors, students and relatives can be constructed on an international scale.

References

[BCFM09] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. Methodologies for data quality assessment and improvement. *ACM computing surveys (CSUR)*, 41(3):1–52, 2009.

[Cor] Bob Coret. Open archives api. "https://www.openarch.nl/api/docs/". Accessed: 2022-07-02.

[DND13] Nikhil Debbarma, Gautam Nath, and Hillol Das. Analysis of data quality and performance issues in data warehousing and business intelligence. *International Journal of Computer Applications*, 79:20–26, 2013.

[Her19] My Heritage. Watch out for genealogy errors. "https://education.myheritage.com/article/watch-out-for-genealogy-errors/", 2019. Accessed: 2022-08-06.

[IMS+20] M. Zafar Iqbal, Ghulam Mustafa, Nadeem Sarwar, Syed Hamza Wajid, Junaid Nasir, and Shaista Siddque. A review of star schema and snowflakes schema. In *Intelligent Technologies and Applications*, pages 129–140, Singapore, 2020. Springer Singapore.

[Lei] Erfgoed Leiden. Ons verhaal. "https://www.erfgoedleiden.nl/werkgebied/organisatie/onze-organisatie". Accessed: 2022-07-28.

[Mar05] Richard Marsh. Drowning in dirty data? it's time to sink or swim: A four-stage methodology for total data quality management. *The Journal of Database Marketing & Customer Strategy Management*, 2:105–112, 01 2005.

[MM] Maurits Meijer and Judith Moortgat. Toelichting a2a datamodel. "https://a2a.coret.org/A2A/A2ABeschrijving_v1.8.pdf". Accessed: 2022-07-30.

[MVDM20] Rick Mourits, Ingrid Van Dijk, and Kees Mandemakers. From matched certificates to related persons. *Historical Life Course Studies*, 9, 2020.

[RMR+20] Joe Raad, Rick Mourits, Auke Rijpma, Ruben Schalk, Richard L. Zijdeman, Kees Mandemakers, and Albert Merono-Penuela. Linking dutch civil certificates. In *Third Workshop on Humanities in the Semantic Web (WHiSe 2020)*, pages 47–58. CEUR-WS, 2020.

[SAZ+19] Manel Souibgui, Faten Atigui, Saloua Zammali, Samira Cherfi, and Sadok Ben Yahia. Data quality in etl process: A preliminary study. *Procedia Computer Science*, 159:676–687, 2019. Knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 23rd International Conference KES2019.

[Sch14] Marijn Schraagen. *Aspects of Record Linkage*. PhD dissertation, Leiden University, 2014.

[Sch22] Rick Schreuder. *Design of a database supporting the exploration of historical documents and linked register data*. Bachelor's thesis, Leiden University, 2022.

[SS10] Ranjit Singh and Kawaljeet Singh. A descriptive classification of causes of data quality problems in data warehousing. *International Journal of Computer Science Issues (IJCSI)*, 7(3):41, 2010.

[vdBvDM+21] Niels van den Berg, Ingrid K. van Dijk, Rick J. Mourits, P. Eline Slagboom, Angelique A. P. O. Janssens, and Kees Mandemakers. Families in comparison: An individual-level comparison of life-course and family reconstructions between population and vital event registers. *Population Studies*, 75(1):91–110, 2021. PMID: 32056500.

A EER Diagram Database Model Rick Schreuder

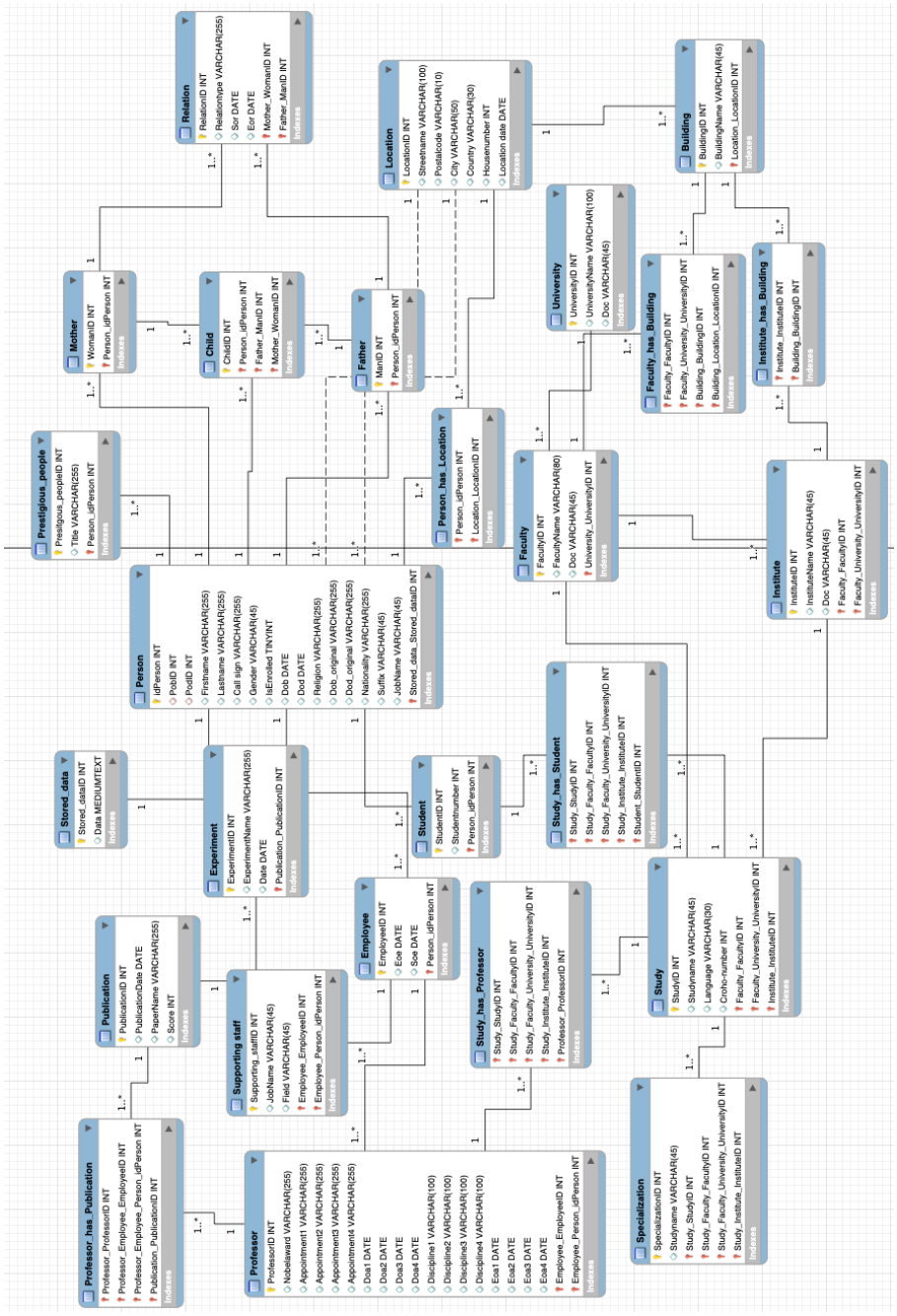

Figure 5: EER diagram database model Rick Schreuder

B Star Schema & Snowflake Schema

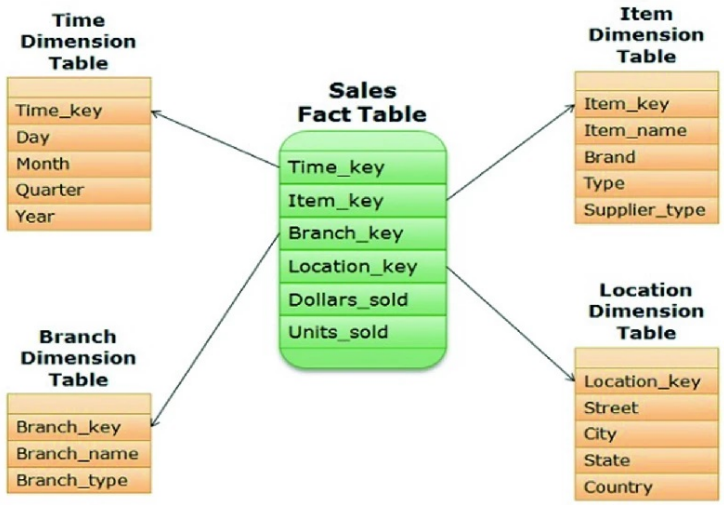

Figure 6: Examples of snowflake schema [IMS+20, Figure 2]

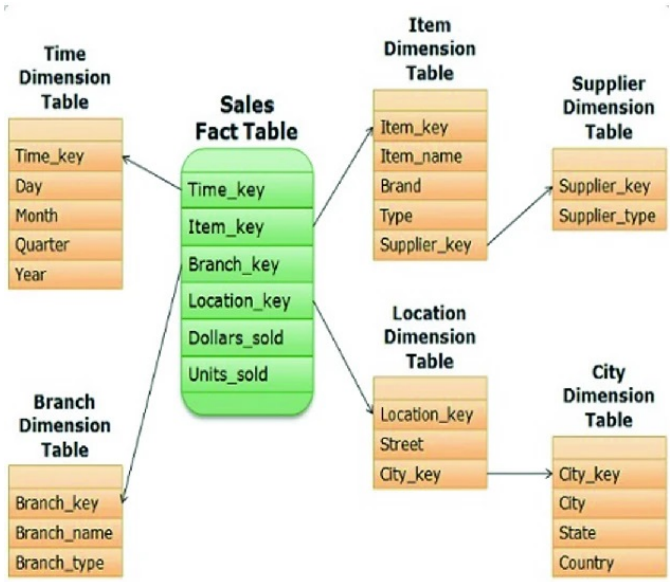

Figure 7: Examples of snowflake schema [IMS+20, Figure 3]
The "supplier dimension table" is an example of a subdimension table.

C Open Archives API Query Responses

C.1 Query 1: show all matching records

First record from result for https://api.openarch.nl/1.0/records/search.json?name=~Boerhaave&archive=elo&number_show=100&sourcetype=DTB%20Trouwen&start=0:


```
{
  "query": {
    "archive": "Erfgoed Leiden en omstreken",
    "sourcetype": "DTB Trouwen",
    "name": "~Boerhaave",
    "only_results_with_scans": false,
    "start": 0,
    "number_show": 1,
    "sort": 1,
    "language": "nl"
  },
  "response": {
    "number_found": 18,
    "docs": [
      {
        "pid": "Person:3a94c7c4-6ff2-2a91-0455-fd7d8e406f03",
        "identifier": "7134512d-1ae2-94ca-cb31-6bd0a7fea6d5",
        "archive_code": "elo",
        "archive_org": "Erfgoed Leiden en omstreken",
        "archive": "Erfgoed Leiden en omstreken",
        "personname": "Bernard Boerhaave van Altena",
        "relationtype": "Eerdere man",
        "_relationtype": "Eerdere man",
        "eventtype": "Trouwen",
        "_eventtype": "Trouwen",
        "eventdate": {
          "day": 20,
          "month": 6,
          "year": 1776
        },
        "eventplace": "Leiden",
        "sourcetype": "DTB Trouwen",
        "url": "https:\/\/www.openarch.nl\/elo:7134512d-1ae2-94ca-
          cb31-6bd0a7fea6d5"
      }
    ]
  }
}
```

C.2 Query 2: get record by identifier

Result for https://api.openarch.nl/1.0/records/show.json?archive=elo&identifier=7134512d-1ae2-94ca-cb31-6bd0a7fea6d5:


```
[
  {
    "a2a_Person": [
      {
        "pid": "Person:dcec5b23-53ad-bf3e-4bc6-bca9f8181577",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Jan"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Vergouw"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        }
      },
      {
        "pid": "Person:5e8ed55f-8335-a9c1-7d3e-3843763c3873",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Gerrit"
          },
          "a2a_PersonNamePrefixLastName": {
            "a2a_PersonNamePrefixLastName": "de"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Zwager"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        }
      },
      {
        "pid": "Person:9847761f-0f99-f0fa-6e60-2d7582384297",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Anthony"
          },
          "a2a_PersonNamePrefixLastName": {
```

```
            "a2a_PersonNamePrefixLastName": "de"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Zwager"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        }
      },
      {
        "pid": "Person:3a94c7c4-6ff2-2a91-0455-fd7d8e406f03",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Bernard"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Boerhaave van Altena"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        }
      },
      {
        "pid": "Person:c95b256f-6606-38aa-238b-66535dac10a3",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Magdalena Maria Elisabeth"
          },
          "a2a_PersonNamePrefixLastName": {
            "a2a_PersonNamePrefixLastName": "de"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Zwager"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        },
        "a2a_Residence": {
          "a2a_Place": {
            "a2a_Place": "Leiden, Langebrugge"
          }
        }
```

```
      },
      {
        "pid": "Person:59739723-c236-f5d2-d958-588bee4f2767",
        "a2a_PersonName": {
          "a2a_PersonNameFirstName": {
            "a2a_PersonNameFirstName": "Jan Willem"
          },
          "a2a_PersonNameLastName": {
            "a2a_PersonNameLastName": "Vergeer"
          }
        },
        "a2a_Gender": {
          "a2a_Gender": "Onbekend"
        },
        "a2a_Residence": {
          "a2a_Place": {
            "a2a_Place": "Leiden, Cellebroersgragt"
          }
        },
        "a2a_BirthPlace": {
          "a2a_Place": {
            "a2a_Place": "Leiden"
          }
        },
        "a2a_Profession": {
          "a2a_Profession": "zilversmit"
        }
      }
    ],
    "a2a_Event": {
      "eid": "Event1",
      "a2a_EventType": {
        "a2a_EventType": "Trouwen"
      },
      "a2a_EventPlace": {
        "a2a_Place": {
          "a2a_Place": "Leiden"
        }
      }
    },
    "a2a_RelationEP": [
      {
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:dcec5b23-53ad-bf3e-4bc6-bca9f
            8181577"
```

```
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "Getuige"
        }
      },
      {
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:5e8ed55f-8335-a9c1-7d3e-38437
            63c3873"
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "Getuige"
        }
      },
      {
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:9847761f-0f99-f0fa-6e60-2d758
            2384297"
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "Getuige"
        }
      },
      {
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:3a94c7c4-6ff2-2a91-0455-fd7d8
            e406f03"
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "other:Eerdere man"
        }
      },
      {
```

```
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:c95b256f-6606-38aa-238b-66535dac10a3"
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "Bruid"
        }
      },
      {
        "a2a_PersonKeyRef": {
          "a2a_PersonKeyRef": "Person:59739723-c236-f5d2-d958-588bee4f2767"
        },
        "a2a_EventKeyRef": {
          "a2a_EventKeyRef": "Event1"
        },
        "a2a_RelationType": {
          "a2a_RelationType": "Bruidegom"
        }
      }
    ],
    "a2a_Source": {
      "a2a_SourcePlace": {
        "a2a_Place": {
          "a2a_Place": "Leiden"
        }
      },
      "a2a_SourceIndexDate": {
        "a2a_From": {
          "a2a_From": "1772-01-01"
        },
        "a2a_To": {
          "a2a_To": "1778-12-31"
        }
      },
      "a2a_SourceType": {
        "a2a_SourceType": "DTB Trouwen"
      },
      "a2a_SourceReference": {
        "a2a_Place": {
          "a2a_Place": "Leiden"
        },
```

```
        "a2a_InstitutionName": {
          "a2a_InstitutionName": "Erfgoed Leiden"
        },
        "a2a_Archive": {
          "a2a_Archive": "1004"
        },
        "a2a_Collection": {
          "a2a_Collection": "Archiefnaam: Nederlands Hervormd
             Ondertrouw (1575-1795), Deel: 44, Periode: 1772-1778"
        },
        "a2a_Book": {
          "a2a_Book": "NH Ondertrouw VV. september 1772 - 1778."
        },
        "a2a_Folio": {
          "a2a_Folio": "VV-178v"
        },
        "a2a_RegistryNumber": {
          "a2a_RegistryNumber": "44"
        }
      },
      "a2a_SourceAvailableScans": {
        "a2a_Scan": [
          {
            "a2a_OrderSequenceNumber": {
              "a2a_OrderSequenceNumber": "1"
            },
            "a2a_Uri": {
              "a2a_Uri": "https:\/\/images.memorix.nl\/lei\/thumb\/
                 640x480\/59e9da52-7439-47b7-e086-13562cd695a1.jpg"
            },
            "a2a_UriViewer": {
              "a2a_UriViewer": "https:\/\/www.erfgoedleiden.nl\/
                 collecties\/personen\/zoek-op-personen\/deeds\/713
                 4512d-1ae2-94ca-cb31-6bd0a7fea6d5"
            },
            "a2a_UriPreview": {
              "a2a_UriPreview": "https:\/\/images.memorix.nl\/lei\/
                 thumb\/250x250\/59e9da52-7439-47b7-e086-13562cd695
                 a1.jpg"
            }
          },
          {
            "a2a_OrderSequenceNumber": {
              "a2a_OrderSequenceNumber": "2"
            },
```

```
            "a2a_Uri": {
              "a2a_Uri": "https:\/\/images.memorix.nl\/lei\/thumb\/
                 640x480\/e2bde67b-0cc9-bbd7-8553-4192e7394211.jpg"
            },
            "a2a_UriViewer": {
              "a2a_UriViewer": "https:\/\/www.erfgoedleiden.nl\/
                 collecties\/personen\/zoek-op-personen\/deeds\/713
                 4512d-1ae2-94ca-cb31-6bd0a7fea6d5"
            },
            "a2a_UriPreview": {
              "a2a_UriPreview": "https:\/\/images.memorix.nl\/lei\/
                 thumb\/250x250\/e2bde67b-0cc9-bbd7-8553-4192e73942
                 11.jpg"
            }
          }
        ]
      },
      "a2a_SourceLastChangeDate": {
        "a2a_SourceLastChangeDate": "2013-02-04"
      },
      "a2a_SourceDigitalOriginal": {
        "a2a_SourceDigitalOriginal": "https:\/\/www.erfgoedleiden.
           nl\/collecties\/personen\/zoek-op-personen\/deeds\/71345
           12d-1ae2-94ca-cb31-6bd0a7fea6d5"
      },
      "a2a_RecordGUID": {
        "a2a_RecordGUID": "{7134512d-1ae2-94ca-cb31-6bd0a7fea6d5}"
      },
      "a2a_SourceRemark": [
        {
          "Key": "Opmerking",
          "a2a_Value": {
            "a2a_Value": "bruidegom is gereformeerd, bruid brengt \
               u00e9\u00e9n kind in<br \/>Datum ondertrouw: 20-06-1
               776\n<br\/><a href=\"\/collecties\/archieven\/
               archievenoverzicht\/search\/list\/withscans\/0\/
               findingaid\/1004\/file\/44\/start\/0\/limit\/10\/
               flimit\/5\">Inventarisnummer 44 van archiefnummer 10
               04 in Archieven<\/a>"
          }
        },
        {
          "Key": "Provenance",
          "a2a_Value": {
            "a2a_Value": "A2Acollection oai-pmh_20211013_1343_00001
```

```
                343.xml van ELO"
            }
        }
    ]
  }
 }
]
```

D Github

The link to the Github repository of my branch: https://github.com/LiacsProjects/linkingUCD_code/tree/adapters-michael. The source code of the adapters and the code for the reproduction of the enhanced database model can be found there.