



Emanuel Hrastrnig BSc.

A LINKED DATA APPROACH FOR DIGITAL HUMANITIES.

Prototypical storage of a dialect data set in a triplestore.

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Geomatics Science

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Dipl.-Ing (FH) Dr.techn. Johannes Scholz

Institute of Geodesy

External Supervisor: Mag. Eveline Wandl-Vogt
(Austrian Academy of Sciences)

Graz, January 2018

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master's thesis.

Date

Signature

ABSTRACT

Despite “Linked Data” not being a new term in digital humanities and bringing advantages in the context of semantics and interoperability to this field of research, existing data often does not get published as linked data. This thesis shows how spatiotemporal data from the relational database of the “Datenbank der bairischen Mundarten in Österreich electronically mapped” project can be stored and queried in the form of RDF triples by creating a spatiotemporal linked data model and providing an efficient use of existing resources and common publishing tools. In the presented approach, the lexical data is stored in a virtual triplestore and linked to spatial RDF data and linked open data from DBpedia and GeoNames. Further on, a proof-of-concept prototype, which is part of a testing environment for using spatiotemporal SPARQL queries and full-text search, is exposed. By testing and analyzing the capabilities of the prototype, the presented Linked Data approach is shown to provide functionality similar to the original relational database approach, but with the benefit of interoperable, semantic and cross-source queries. It also shows that lexicographic projects can benefit significantly from this model as it is an easy way to publish their spatiotemporal data as linguistic inked open data.

ZUSAMMENFASSUNG

Obwohl “Linked Data” kein neuer Begriff im Bereich der digitalen Geisteswissenschaften ist und der Einsatz dieser Technologie Vorteile im Bezug auf Semantik und Interoperabilität mit sich bringt, werden vorhandene Daten oft nicht als Linked Data veröffentlicht. Diese Masterarbeit zeigt wie räumlich-zeitliche Daten aus der relationalen Datenbank des Projektes “Datenbank der bairischen Mundarten in Österreich electronically mapped” in Form von RDF Triples gespeichert und abgefragt werden können. Dazu wird ein räumlich-zeitliches Linked Data Modell verwendet, bei dem die effiziente Nutzung vorhandener Ressourcen und gängiger Publishing-Tools im Vordergrund steht. Im vorgestellten Ansatz werden die lexikalischen Daten in einem virtuellen Triplestore gespeichert und mit räumlichen RDF-Daten sowie Daten von “DBpedia” und “GeoNames” verknüpft. Weiteres wird ein Proof-of-Concept-Prototyp präsentiert, der Teil einer Testumgebung ist, die speziell für die Verwendung von räumlich-zeitlichen SPARQL Abfragen und Volltextsuche ausgelegt ist. Durch Testen und Analysieren der Fähigkeiten des Prototyps wird gezeigt, dass der präsentierte Linked Data Ansatz eine ähnliche Funktionalität wie der ursprünglichen Ansatz der relationalen Datenbank liefert, aber den Vorteil von interoperablen, semantischen Daten und quellübergreifenden Abfragen mit sich bringt. Außerdem wird gezeigt, dass lexikographische Projekte erheblich von diesem Modell profitieren können, da es eine einfache Möglichkeit ist, ihre räumlich-zeitlichen Daten als “Linguistic Linked Open Data” zu veröffentlichen.

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisor Ass.Prof. Dipl.-Ing (FH) Dr.techn. Johannes Scholz, who made it possible for me to work on this topic. I very much appreciate that he has always taken the time to answer my questions and to help me with constructive suggestions. I would also like to thank my second supervisor, Mag. Eveline Wandl-Vogt, who was my contact person at the Österreichische Akademie der Wissenschaften and without whom this thesis would not have been possible. A big thanks goes to my family and friends who have always supported me, making them an important part of this thesis as well. Finally, I would like to thank my fellow students and the academic staff of the TU Graz for making these years of study a wonderful and memorable time.

CONTENTS

List of Abbreviations	xi
1 Introduction	1
1.1 Motivation	1
1.2 Goals	3
1.3 Approach	4
1.4 Literature	5
2 Fundamentals of the Semantic Web	8
2.1 Definitions	8
2.1.1 Semantic Web	8
2.1.2 Linked Data	9
2.1.3 Ontology in Linked Data	10
2.1.4 Digital Humanities and Geographic Information Science	11
2.2 Semantic Web Technologies	12
2.2.1 Uniform Resource Identifier	12
2.2.2 Extensible Markup Language	12
2.2.3 Resource Description Framework	13
2.2.4 Ontology Language	17
2.2.5 Language Overview	18
2.2.6 Linked Data query language	19
2.3 Publishing Linked Data	22
2.3.1 Choosing Uniform Resource Identifiers	23
2.3.2 Defining Vocabulary	24
2.3.3 Link Generation	27
2.3.4 Metadata	28
2.3.5 Publishing Tools	28
2.3.6 Additional Approach	30
2.4 Linked Open Data	30
2.5 Linguistics and spatiotemporal Linked Data	31
3 Approach	32
3.1 Basic Conditions	32
3.1.1 Software	33
3.1.2 Standards	33
3.2 Linked Data publishing	34
3.2.1 Resource identification	34

3.2.2	Vocabulary creation	34
3.2.3	Link generation	34
3.2.4	Usage of Publishing Tools	35
3.2.5	Metadata	36
3.3	Linked Data Exploration	37
3.3.1	Queries	37
3.3.2	Testing Environment	38
4	Prototyping	39
4.1	Initial situation	39
4.1.1	Initial Data	40
4.1.2	Implementation	42
4.2	Choosing Uniform Resource Identifiers (URIs)	43
4.3	From a relational database to an ontology	44
4.3.1	Providing the Ontology	45
4.4	Data preparation	46
4.4.1	Data quality	46
4.4.2	Linked Open Data	48
4.5	Data mapping	53
4.5.1	Mapping Language	54
4.5.2	Spatial data	58
4.5.3	Lexical data	59
4.6	Creation a testing environment	59
4.6.1	Relational database management system	60
4.6.2	Ontology Editor	60
4.6.3	Mapping Tool	61
4.6.4	Spatial triplestore	61
4.6.5	Linked Data front end	62
4.6.6	Web server	62
4.6.7	Overview	63
4.7	Linked Data applications	63
4.7.1	Query Tool	64
4.7.2	Search Tool	65
5	Results of the project	67
5.1	Prototype	67
5.1.1	Ontology	67
5.1.2	Mapping Server	68
5.1.3	Spatial triplestore	68
5.1.4	Query Tool	69
5.1.5	Search Tool	71
5.2	Interpretation	74
5.2.1	Capability of the Linked Data prototype	74
5.2.2	Influence on digital humanities	75

6 Summary and outlook	76
6.1 Summary	76
6.2 Outlook	77
Bibliography	78
Appendix	81

LIST OF FIGURES

2.1	Linking Open Data cloud diagram 2014, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak (http://lod-cloud.net/).	10
2.2	Example for a graph describing the Olympic Games.	15
3.1	Diagram of the approach.	32
3.2	A graph representing a location with its spatial information stored on a separate source.	35
3.3	Example for a graph handling a time period.	36
3.4	Query approach.	37
4.1	A voucher describing a bottle crate.	40
4.2	Spatial tables in the “Datenbank der bairischen Mundarten in Österreich electronically mapped” (dboe@ema).	41
4.3	Tables with temporal information.	42
4.4	Workflow.	43
4.5	A part of the dboe@ema ER-Diagram in Chen-Notation.	44
4.6	Faceted variation of data describing the year of publication.	47
4.7	Example for spatial dboe@ema data.	49
4.8	Example for geonames links stored in the relational dboe@ema database.	52
4.9	Example for DBpedia links stored in the relational dboe@ema database.	53
4.10	Simplified data structure of the dboe@ema Linked Data approach.	60
4.11	Simplified system structure of the dboe@ema Linked Data approach.	63
4.12	Simplified structure of the application for querying the triplestores.	64
4.13	Simplified structure of the application for searching and showing linked dboe@ema data.	66
5.1	Screenshot of a resource, represented by the D2R-Server Hypertext Markup Language (HTML) interface.	68
5.2	Screenshot of an example query showing the first five triples, stored in the Strabon triplestore.	69
5.3	Screenshot of a geospatial resource, represented by the enhanced Pubby interface.	70

5.4	Demonstration of the query tool (part 1).	70
5.5	Demonstration of the query tool (part 2).	71
5.6	Demonstration of the search tool.	72
5.7	Geodata accessed through the search tool.	73
5.8	Geonames resource accessed through the search tool.	73

LIST OF ABBREVIATIONS

API	Application Programming Interface
CRS	Coordinate Reference System
CURL	Client for URLs
DARIAH	Digital Research Infrastructure for the Arts and Humanities
DBMS	Database Management System
DBMS	Database Management System
dboe@ema	“Datenbank der bairischen Mundarten in Österreich electronically mapped”
ERM	Entity Relationship Model
FOAF	Friend of a Friend
GIScience	Geographic Information Science
GIS	Geographic Information System
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ID	Identifier
IRI	Internationalized Resource Identifier
ITU	International Telecommunication Union
JDBC	Java Database Connectivity
JSON	JavaScript Object Notation
LOD	Linked Open Data
OGC	Open Geospatial Consortium
OWL	OWL Web Ontology Language
PHP	PHP: Hypertext Preprocessor

PURL	Persistent Uniform Resource Locator
RDFS	RDF Schema
RDF	Resource Description Framework
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium (http://www.w3.org/)
WGS84	World Geodetic System 1984
WKT	Well Known Text (as defined by Simple Features or ISO 19125)
WWW	World Wide Web
XML	Extensible Markup Language

CHAPTER 1

INTRODUCTION

1.1 MOTIVATION

According to the broadband report ¹, which is published by the UNESCO and the International Telecommunication Union (ITU) every year, there were estimates, that nearly 3.5 billion people were online by the end of 2016. This means that nearly half of the world's population is already using the Internet from time to time. When we talk about the Internet, first of all, we think about the “Web of Documents”. This Web, that evolved at the end of the 20th century, has its focus on publishing documents and make them readable and accessible to the public. The documents have to fulfill standards and recommendations so that a web browser can interpret them and present their content to the user. The Web of Documents was designed to be used by human beings. Documents can be linked to each other, but context is only given to the reader, who can interpret the content semantically (Hall & Tiropanis, 2012).

In the early 21st century, Tim Berners-Lee, a founder of the World Wide Web (WWW), had a vision about the “Semantic Web”, where all the data is linked and includes semantics. This should enable machines to interpret the given data and understand its meaning, making them so called “agents” that browse the Semantic Web and complete tasks for the users (Tim Berners-Lee, Hendler, & Lassila, 2001). Thereby, the data is served in a unified model, that expresses all things and its relationships to so called “triples”. A triple, however, consists of subject, predicate and object. These triples are recorded in triplestores. Although this technology is still very young, it has yet established in many fields of research by bringing new opportunities in exploring and analyzing data.

According to Kuhn, Kauppinen, and Janowicz (2014), Geographic Information Science (GIScience) is already an important part of the Semantic Web and Linked Data. It is also pointed out that a lot of spatial data can be found in the Linked Data cloud, allowing it to enhance nearly every data with spatial information by just linking it. Well established standards

¹<http://www.broadbandcommission.org/publications/Pages/SOB-2016.aspx>

and functionalities that are known from Geographic Information Systems (GIS) got ported for the usage within the Semantic Web and Linked Data. So there are spatial data-stores as well as spatial query languages. This is a huge progress in terms of interdisciplinary research and a chance for GIScience researchers to get even more established in Linked Data.

This thesis will try to bring Linked Data closer to digital humanities with a slight focus on spatiotemporal data. To achieve this a lexical dataset stored in a relational database gets published as Linked Data. The initial data originates in the “Datenbank der bairischen Mundarten in Österreich elektronisch gescannt” (dboe@ema) project, which was implemented from 2007 to 2010. Beneath the lexical data, this relational database also distributes spatial information and handles spatial requests. The model of the database, the so called relational model, consists of tables that contain the entities and their attributes. This structure makes the model very inflexible when it comes to an extension.

Interoperability and federation are two of the strengths of Linked Data. The underlying model of stored statements, which consist of subject, predicate and object allows it to link data without the need to change the model. This will enable to link the lexical data to spatial data, temporal data and all kinds of Linked Open Data (LOD). Hence, storing the data in a triple-store will make the corpus more suitable for future developments in the field of Semantic Web (Chiarcos, Hellmann, & Nordhoff, 2011).

When dividing the topic into a more theoretical and a more practical part, following questions will emerge:

- What is a suitable Linked Data approach for the lexical dboe@ema data?
- How does its functionality compare to the relational database system?
- What are the benefits for the dboe@ema project of implying this approach?

The answers to the questions will be given in this thesis, which is structured in six chapters. While the first chapter contains the introduction, the second chapter tries to bring Semantic Web technologies closer to the reader. It is important to start with the basics in order to explain modern technologies and approaches that are used by the Linked Data community. Through showing various examples, the attempt of conveying knowledge of the used syntax notations and data models is made. The third chapter explains how technologies can be used to answer the research questions asked

beforehand. Approaches that are introduced in theory in the previous chapter are described in detail and it is shown how they are adjusted to fit this very project. This will not be an attempt to go too far on technical details, as these are discussed in chapter four, the implementation of a prototype. The task of prototyping starts by describing the initial situation regarding data and its structure and is followed by the presentations of a workflow that leads to the prototype. The detailed approach, the used software packages and their configuration are explained in detail. Chapter five contains the results of this thesis. The prototype and its functionalities are highlighted and the answers to the scientific questions are explained. In the last chapter, a summary of the project is given by interpreting the approach and the results and pointing out the answers to the scientific questions. This is followed by an outlook that describes further actions following this project, what could be done in the areas of application and what improvements would make sense.

1.2 GOALS

The main goal of this thesis is to publish the dialect data of `dboe@ema` as Linked Data. This should help to connect the `dboe@ema` data with other dialect datasets. For this purpose, an approach must be found that uses the existing resources in the best way. This means that an attempt to integrate the relational database, in which the initial data is stored, is made. In addition, special attention must be paid to the spatial and temporal parts of the data, since spatiotemporal queries are an important part of this research. Moreover chosen parts of the data should be referenced with data from the LOD cloud. This could be applied to the names of locations or the meanings of dialect words. Furthermore, the approach should be easy to apply and expandable in order to be flexible when it comes to facing changes later on in the project.

Creating an ontology for `dboe@ema` data is also one of the goals. Just as with the approach, the initial resources should be used here as well as possible. The existing database model can be used as a starting point for the creation of the ontology. At this point, the spatial and temporal attributes of the data must be integrated.

In order to verify the feasibility of the approach, a prototype is developed. Furthermore, the prototype should be used to check whether the approach provides the required functionality. This ranges from the incorporation of Linked Open Data to the possibility of spatiotemporal querying the data. In the implementation of the prototype it should be ensured that common standards of the W3C and software of the Linked Data community are used.

This will guarantee high interoperability. Additionally, it is desirable that the prototype will be integrated into the existing workflow of “exploreAT!” (exploring austrias culture through the language glass), the current follow up project of dboe@ema.

The prototype forms the basis for a test environment. This includes applications intended to simulate different use cases. The applications should offer the possibility of spatiotemporal queries as well as the option to search and explore the Linked Data. Attention should also be paid to the visual representation of the results. In this case, a graphical processing of the spatial information.

Furthermore, the influence of the developed approach on the dboe@ema project has to be determined. The aim is to find out how the new approach compares to the existing system, what the benefits are and what future steps can be taken.

1.3 APPROACH

For this project, an approach is chosen, which makes it possible to publish existing data from relational databases as Linked Data. Attention is paid to the peculiarities of spatial data and to adapt the approach from the literature and the community accordingly. The presented approach is a customized version of the approach by Bizer, Heath, and Berners-Lee (2009) and the detailed version can be found in chapter 3. Summarized, it can be said that the approach consists of the following points:

Basic Conditions: The aim here is to ensure that certain guidelines and recommendations are complied during the realization of the project. This is done in order to achieve a certain degree of interoperability and topicality to obtain data which is suitable for the Linked Data cloud. Common standards of the World Wide Web Consortium (<http://www.w3.org/>) (W3C) and Open Geospatial Consortium (OGC) are to be used for this purpose. Additionally, it is focused to use open-source software, because it can be highly customized due to its open source code and licensing.

Ontology: Linked Data needs a proper ontology to bring semantics to the data and make it interpretable for humans and machines. In this case an approach is chosen which uses the relational database structure and maps it to an ontology. Thereby the Entity Relationship Model (ERM) serves as the basis for this step in which tables are converted into classes and the relationships as well as the literal values of the

tables are interpreted as properties. This allows a simple and tailored creation of a vocabulary for this prototype.

Link generation: This step can also be seen as part of data preparation because it precedes the actual data publishing approach. It tries to extend the existing data from the relational database through finding counterparts in the LOD cloud. These matches are stored as links in the database. Later, these links get mapped and published as Linked Data, along with the rest of the data. This ensures that the individuals can be part of the LOD cloud.

Linked Data publication: This is one of the main tasks of the presented approach. It is about using tools to convert and publish the data from the relational database as Linked Data. Thereby the spatial characteristics of the data must be taken into account. Due to technical limitations, the data needs to be divided into a spatial and a lexical part and be stored in two different triplestores. The static, spatial data is stored in spatial triplestore, while the lexical data gets mapped as virtual triplestore in real time. This approach guarantees absolute, spatiotemporal capabilities by using the existing resources in the best way. It also brings high flexibility in maintenance since the static spatial data stays untouched, while the relational database with the lexical data can be updated easily and automatically gets mapped as Linked Data.

Prototyping and testing: For testing purposes, it is inevitable to create a prototype and demonstrate the possibilities of this approach. The prototype should be able to store and query lexical Linked Data as well as providing the possibility to explore the data. In doing so, applications are created which provide a way to test and demonstrate the capabilities of the prototype. In this thesis, two web applications are shown. These applications are a query tool and a search tool for exploring the Linked Data. They are used to compare prototype with the dboe@ema system and to find answers to the scientific questions.

In addition to these very subject-specific points, it is important to mention generally important information such as the literature study, data preparation, the evaluation of the results and the documentation.

1.4 LITERATURE

This section gives a short impression of the literature that is used in this thesis. At the beginning, some basic information about the Semantic Web

is needed. While the article by Tim Berners-Lee et al. (2001) explains the elementary vision behind the Semantic web and its application, the paper by Burgos (2011) provides Semantic Web standards, such as Resource Description Framework (RDF) and OWL Web Ontology Language (OWL), the current semantic languages. Also, the Semantic Web is introduced along with its related vocabularies, which enable the construction of the Semantic Web and how the Linked Data is developed and deployed. Additional Information on the Semantic Web is given in the book by Hitzler, Krötzsch, Rudolph, and Sure (2008). It offers the fundamentals of the Semantic Web in a comprehensible manner and enables a simple and fast entry into the technologies of the Semantic Web. The main focus is on modeling and logical-semantic foundations of Semantic Web technologies. This is provided by a clean separation between an intuitive execution on the one hand and the explanation of formal and theoretical backgrounds on the other. An introduction to OWL and a description the features of each of the sublanguages of OWL is given in the paper by McGuinness and Van Harmelen (2004).

As Linked Data is part of the Semantic Web and an essential component of this project, a widespread knowledge about this topic is needed. The paper by Bizer et al. (2009) contains a lot of information about Linked Data and is a frequently used literature in this thesis. It presents the concept and technical principles of Linked Data and situates these within the broader context of related technological developments. Further, the progress in publishing Linked Data on the Web is shown and applications that have been developed to exploit the Web of Data are reviewed.

Another crucial part of the thesis is the topic of ontologies. Basic information about the universe of discourse and the creation of ontologies is given by Gruber (1993) and Musen (1992). Horrocks, Patel-Schneider, and van Harmelen (2003), on the other hand, focuses more on the philosophy and features of OWL and how it is traced back to older formalisms. A more practical view on ontologies is given by Myroshnichenko and Murphy (2009) as they give a solution on mapping well-formed ERMs to semantically equivalent OWL ontologies, which is one of the key points in this thesis.

A requirement for the prototype is the capability of spatial queries. The technology to accomplish this is GeoSPARQL Protocol and RDF Query Language (SPARQL). The paper by Battle and Kolas (2012) shows overall state of geospatial data in the Semantic Web with a focus on the OGC standard GeoSPARQL and its attempts to unify data access for the geospatial Semantic Web. Therefore, it describes the motivation for GeoSPARQL, the current state of the art in industry and research and gives an example use case and the implementation of GeoSPARQL in a triplestore. Perry and Herring (2012) define the OGC standard of GeoSPARQL and give all the needed details for application.

Spatiotemporal Linked Data, in general, is handled by Kuhn et al. (2014). Though the innovations of Linked Data in context of GIScience are explained and demonstrated with examples. It refers to the impact of Linked Data in GIScience and upcoming challenges.

Geolinguistics is also a part of the project because of its heritage in the dboe@ema system. Chambers and Trudgill (1998) defined Geolinguistics as a field, that researches the geographic distribution of language or its constituent elements by working with digital language databases and the visual exploration of linguistic data. Additional information on Geolinguistics in the context of GIScience is given by Hoch and Hayes (2010), Sibling, Weibel, Glaser, and Bart (2012) and Scholz, Lampoltshammer, Bartelme, and Wandl-Vogt (2016). A description of the dboe@ema project, in particular, is given by Bartelme et al. (2016).

LOD in the context of DBpedia is described by Auer et al. (2007). They present the current status of interlinking DBpedia with other open datasets on the web and outline how DBpedia could serve as a nucleus for an emerging web of open data. Furthermore Chiarcos et al. (2011) show how linguistic LOD can be created and published. They argue that “Open Data” has become very important in a wide range of field and propose the use of Linked Data principles to enable language resources to be published and interlinked openly on the Web. It is also shown that modeling and publishing language resources as Linked Data offers crucial advantages as compared to existing formalisms. In particular, it is explained how this can enhance the interoperability and the integration of linguistic resources. Further benefits of this approach and recent community activities are described in this paper.

Additional to the mentioned literature, there are several web documents released by the W3C and manuals provided software projects which are used in this thesis.

CHAPTER 2

FUNDAMENTALS OF THE SEMANTIC WEB

In this chapter, some basics are provided and an fundamental overview of the used techniques is given. A more precise description of the approach will be discussed in chapter 3.

2.1 DEFINITIONS

2.1.1 *Semantic Web*

The idea of the Semantic Web came up in the late 1990s and was a future vision from the World Wide Web Consortium (<http://www.w3.org/>) (W3C) director Tim Berners-Lee on how the Web could evolve in order to become more accessible for machines and solve more complex tasks for the users. According to Tim Berners-Lee et al. (2001), the majority of the content on the Internet is made to be read and understood by humans and not by computer programs. The web browser can only interpret the layout, links and methods but has no proper capability to handle the semantics. The Semantic Web is an extension to the existing World Wide Web (WWW). This is accomplished by bringing a well-defined meaning to the content of normal Web pages. Therefore, it is possible that so called agents can crawl Web pages and understand their content without the need of artificial intelligence.

In the WWW we can link everything. Thereby the kind of information, which it is linked to, doesn't matter. This attribute gives the WWW a mighty universality. The goal of the Semantic Web is to change, from an Web full of linked documents for people, to a Web full of Linked Data and information, which can be processed by computers. Doing so, the Semantic Web will be decentralized like the known Internet. This characteristic means that the Web can grow very fast but with a lack of consistency (Tim Berners-Lee et al., 2001).

A similar definition is given in the more recent paper by Burgos (2011). Here, the Semantic Web is also considered as an extension of the Web of

Documents. It is claimed to become the leading future technology and it will provide a better understanding and cooperation between people and machines, so that a more intuitive and dynamic Web is possible. A basic requirement for this is that the information in the Semantic Web is well defined in a machine-readable format. This will generate data that can be understood by machines and humans. Considering the Web and its information nowadays, it can be seen that it consists of a lot of documents which makes it hard to find that piece of information that the client is searching for. The solution to this problem will be provided by the Semantic Web and its meaningful data.

The Semantic Web is still in its beginning and these Web-crawling software agents are, furthermore, a future vision. However, with upcoming Linked Open Data (LOD) provided by governments and the open source community, it should already be paid attention to nowadays.

2.1.2 Linked Data

According to Bizer et al. (2009), the term Linked Data defines data which is linked to each other independent from its heritage or source. Possible sources could be two databases maintained by two different organizations or other data that has not been interoperated yet. From a technically point of view, Linked Data has four important attributes:

- It is published on the Web.
- It is machine-readable.
- Its meaning is explicitly defined.
- It is linked to other datasets.

Figure 2.1 shows a diagram of LOD in 2014. The closer to the middle and the bigger the circle is, the more important a dataset is relating to the “Web of Data”. The yellow bubbles represent datasets with “geospatial-background”, which already have a very strong presence.

The given definition showed that Linked Data gives us the opportunity to link every kind data to spatial information with ease. This is also an enormous step for Geographic Information Science (GIScience) to bring spatial information and technologies to other scientific fields of research like digital humanities.

In Linked Data, every dataset consists of things. These things have a distinct Uniform Resource Identifier (URI) and are called resources. For

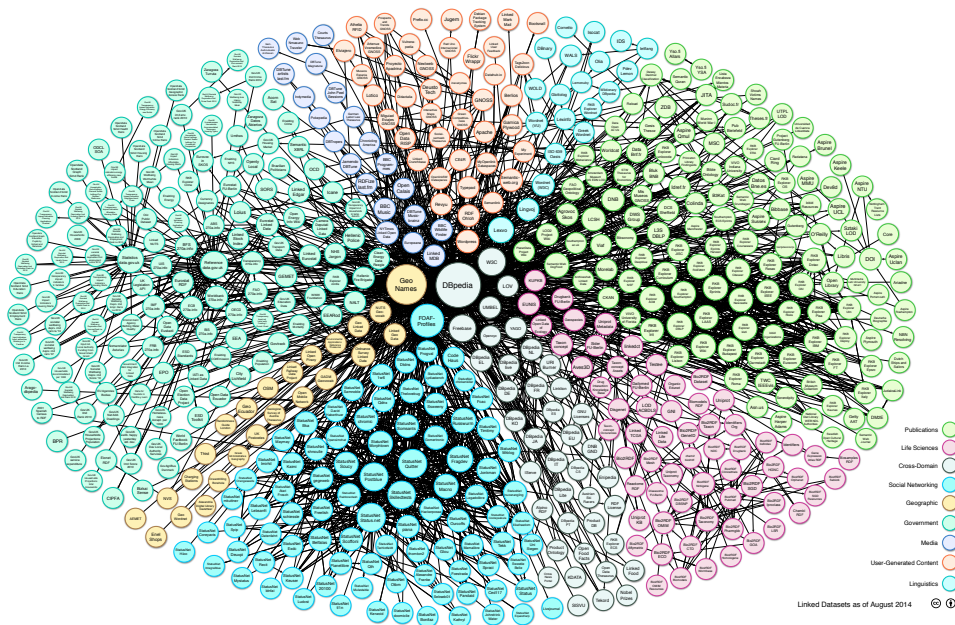


Figure 2.1: Linking Open Data cloud diagram 2014, by Andrejs Abele, John P. McCrae, Paul Buitelaar, Anja Jentzsch and Richard Cyganiak (<http://lod-cloud.net/>).

example, if we have a dataset about places in Austria, each place would be a resource with an own URI (<http://uri/resource/place0123>). This URI does not point to a Hypertext Markup Language (HTML) document like on the normal Internet. This URI points to this specific resource and all its properties and links to other resources and datasets. This creates the possibility to explore more information related to this specific place such as its name, its administrative level, its location and its links to DBpedia¹ or other Linked Data knowledge graphs. In its most basic form, Linked Data is represented by so called triples. These triples, however, consist of a subject, a predicate and an object. Each of these three elements has a URI. This data structure will be explained in detail in section 2.2.

2.1.3 *Ontology in Linked Data*

In philosophy, the term ontology describes the semantic characteristics of existence. According to knowledge-based systems, the set of existing objects belonging to a specific domain is called the universe of discourse. The universe of discourse and its containing relationships between its objects

¹<http://wiki.dbpedia.org/>

are described in a representational vocabulary. The specification of this vocabulary, consisting of definitions for classes, relations, functions and other objects, is called an ontology (Gruber, 1993).

According to Musen (1992), ontologies are a good way to share and reuse knowledge across different software environments. They are not only understandable for humans, but also for computer systems. Horrocks et al. (2003) gave a simple example for an ontology dealing with pizza. In such an ontology Mozzarella and Gorgonzola are defined as a kind of cheese and cheese is a vegetarian topping. With this information a software could interpret a pizza topped with Mozzarella and Grogonzola as a “vegetarian pizza”.

2.1.4 Digital Humanities and Geographic Information Science

Burdick, Drucker, Lunenfeld, Presner, and Schnapp (2012) argue that digital humanities developed from their classical and early modern precursors. Some disciplines that make up the modern humanities are literature, philosophy, classics, rhetoric, history, and studies of art, music, and design. They help to define culture and gain a greater understanding of the human experience. Jannidis, Kohle, and Rehbein (2017) say that we can understand digital humanities as the sum of all attempts to apply information techniques to the subject of humanities.

In Jessop (2008) it is pointed out that spatial-temporal information is an important part of humanities as it shows where people live, events occur and how humanity evolved over time. It is also mentioned that the usage of the Geographic Information System (GIS) in digital humanities have changed over time from a quantitative usage in form of simple maps to a more qualitative use by generating complex interactive maps with multiple data sources and analysis tools. All in all, it can be said that GIScience already plays an important role in digital humanities and will bring new possibilities to this scholarship in the future.

In larger digital humanities infrastructures, GIScience and its technologies have already arrived. An example for such an infrastructure is the Digital Research Infrastructure for the Arts and Humanities (DARIAH). It is a European project, in which scholars from arts and humanities are working with computational methods. Within the DARIAH community, a working group for “geohumanities” is on the rise.

According to Dear, Ketchum, Luria, Richardson, et al. (2011), the term geohumanities refers to the rapidly growing field of the interaction between geography and humanities. The boundaries of these classical disciplines are

broken by research and practices driven by social, technological and political issues. So it happens that geography is now also concerned with humanities and in return, place and space are increasingly incorporated in humanities. With the approach of these two disciplines, new subject areas which require a transdisciplinary perspective and a combination of methods, emerge.

2.2 SEMANTIC WEB TECHNOLOGIES

2.2.1 *Uniform Resource Identifier*

As described by T. Berners-Lee, Fielding, and Masinter (2005), a URI provides a simple and extensible means for identifying a resource. It is a compact sequence of characters that identifies an abstract or physical resource. URIs have a global scope and are interpreted consistently regardless of context, though the result of that interpretation may be in relation to the end-user's context. For example, `http://localhost/` has the same interpretation for every user of that reference, even though the network interface corresponding to "localhost" may be different for each end-user. However, an action made on the basis of that reference will take place in relation to the users context, which implies that an action intended to refer to a globally unique thing must use a URI that distinguishes that resource from all other things.

According to Linked Data, Hypertext Transfer Protocol (HTTP) URIs should be used as names for things so that they can be looked up by people. This is one rule out of four defined by Tim Berners-Lee (2006) when it comes to publishing Linked Data.

For example, the URI `http://dbpedia.org/resource/Graz` points a dataset dealing with the city of "Graz", stored by the DBpedia² project.

The term Internationalized Resource Identifier (IRI) should also be mentioned. It extends the classical URI character set with characters from the Universal Character Set (ISO 10646).

2.2.2 *Extensible Markup Language*

Markup languages are used in computer science to provide additional information to certain parts of text documents. It is also a matter of annotating these parts of the document. The information added by such annotations is

²<http://wiki.dbpedia.org>

commonly called metadata (data that describes other data). In Extensible Markup Language (XML), tags are used for labeling and setting the logical structure of a document. Data in XML documents can be distributed and processed by machines, so XML is a kind of data exchange format. XML is an important, basic technology for creating structured documents, especially through the uniform and standardized way of creating trees and attribute-value pairs. In addition, XML as a meta-language allows you to define your own markup languages. So XML is a standardized, widely used meta-language, which gives us machine-readability. However, from the perspective of the Semantic Web, XML tags are basically not more efficient than the natural language, because they are only words that can be ambiguous and whose relationships are not uniquely defined. Therefore, tags can have a meaning for people, but for machines they are still meaningless in the sense of “without semantics”. The strength of XML, the universal way of exchanging data from any text document, is also its weakness. It is not possible to encode the meaning of annotations in a way that allows machine-sided processing up to the automatic derivation of not explicitly given knowledge. Therefore, XML from the perspective of the Semantic Web is primarily used as a basis for defining the other languages like Resource Description Framework (RDF)(S) and OWL Web Ontology Language (OWL) (Hitzler et al., 2008).

The XML Code seen in listing 2.1 will be interpreted by humans as a description of the two Olympic Summer Games in 2012 and 2016 but has no semantic meaning to a machine.

Listing 2.1: XML xample.

```
<olympicgames>
  <summergames>
    <hostcity>Rio de Janeiro</hostcity>
    <year>2016</year>
    <athletes>11237</athletes>
  </summergames>
  <summergames>
    <hostcity>London</hostcity>
    <year>2012</year>
    <athletes>10520</athletes>
  </summergames>
</olympicgames>
```

2.2.3 *Resource Description Framework*

According to Schreiber, Raimond, Manola, Miller, and Brian (2014), the RDF is a framework for expressing information about resources, in which resources can be anything, including documents, people, physical objects and abstract concepts. It is intended for situations in which information on the

Web needs to be processed by applications rather than being only displayed to people. Therefore, it provides a common framework for expressing this information, in order to be exchanged between applications without loss of meaning. Since it is a common framework, application designers can leverage the availability of common RDF parsers and processing tools. The ability to exchange information between different applications means that the information may be made available to applications other than those for which it was originally created. In particular, RDF can be used to publish and interlink data on the Web. It is also mentioned that RDF allows to make statements about resources. These statements consist of three elements which are called triples. They have a the simple structure in the form of <subject> <predicate> <object>. Such a RDF statement expresses a relationship between a subject and an object. The predicate represents the nature of their relationship. This directional relationship is called a property. They are called triples due to the fact that RDF statements consist of three elements. The same resource is often referenced in multiple triples. The ability to have the same resource be in the subject position of one triple and the object position of another makes it possible to find connections between triples, which is the idea behind Linked Data. As described by Schreiber et al. (2014), triples can consist of the following three types of data:

IRI : IRIs can appear in all three positions of a triple. They are used to identify resources. The notion of IRI is a generalization of URI, allowing non-ASCII characters to be used in the IRI character string.

Literal : Literals are basic values that are not IRIs. Literals may only appear in the object position of a triple. This includes many data types defined by XML Schema such as string, boolean, integer, decimal and date.

Blank Node : Blank nodes can appear in the subject and object position of a triple. They can be used to denote resources without explicitly naming them with an IRI.

For the sake of simplicity, all the following examples are shown in the turtle syntax. Turtle³ and N-Triples⁴ are subsets of of N3⁵ that provide a short notation and are easy to read for humans. In order to save space, the used prefixes are listed in listing 2.2.

³<https://www.w3.org/TR/turtle/>

⁴<https://www.w3.org/TR/n-triples/>

⁵<https://www.w3.org/TeamSubmission/n3/>

Listing 2.2: prefix example.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>.
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>.
@prefix owl: <http://www.w3.org/2002/07/owl#>.
@prefix xsd: <http://www.w3.org/2001/XMLSchema#>.
@prefix ex1: <http://www.example.org/olympicgames#>.
@prefix ex2: <http://www.example.org/geodata#>.
@prefix geo: <http://www.opengis.net/ont/OGC-Geo\ac{SPARQL}/1.0/>.
@prefix geosf: <http://www.opengis.net/def/dataType/OGC-SF/1.0/>.
@prefix geof:
  <http://www.opengis.net/def/queryLanguage/OGC-Geo\ac{SPARQL}/1.0/function/>.
```

The graph shown in figure 2.2 describes five triples concerning the Olympic Summer Games in Rio de Janeiro in 2016 and some of its attributes such as its year, athletes and host-city. This graph can be serialized using RDF. Listing 2.3 shows the graph in turtle syntax.

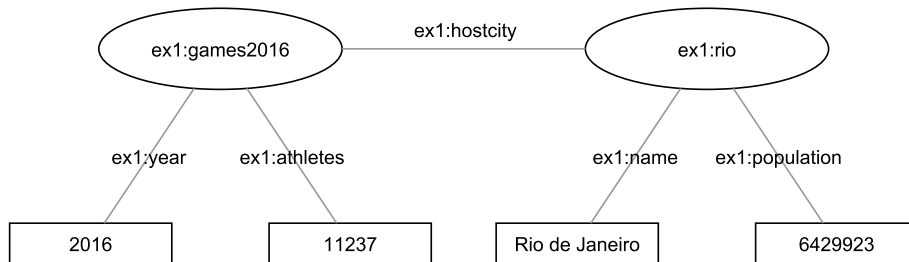


Figure 2.2: Example for a graph describing the Olympic Games.

Listing 2.3: RDF example.

```
ex1:games2016
  ex1:year "2016" ;
  ex1:hostcity ex1:rio ;
  ex1:athletes "11237" ;

ex1:rio
  ex1:name "Rio_de_Janeiro" ;
  ex1:population "6429923" ;
```

As already mentioned, RDF is a data model that provides a framework to publish Linked Data on the Web. When it comes to semantics, RDF lacks the possibility to create meaningful ontologies for the data. This is where RDF Schema (RDFS) is constituted. According to Brickley, Guha, and McBride (2014), RDFS provides a data-modeling vocabulary for RDF data and is a semantic extension of RDF. It provides mechanisms for describing groups of related resources and the relationships between these resources. RDFS is written in RDF and provides terms as resources that are used to determine characteristics of other resources such as the domains and ranges of properties. Resources may be divided into groups called classes. The members of a class are known as instances of the class. Classes are resources

themselves. They are often identified by IRIs and may be described using RDF properties. The `rdf:type` property may be used to state that a resource is an instance of a class. All things described by RDF are called resources, and are instances of the class `rdfs:Resource`. This is the class of everything. All other classes are subclasses of this class. `rdfs:Resource` is an instance of `rdfs:Class`. RDFS also gives the concept of sub property. The `rdfs:subPropertyOf` property may be used to state that one property is a sub property of another. All resources which are related by a property are also related by its sub property. Properties can have a range and a domain. A `rdfs:domain` is used to state that any resource that has a given property is an instance of one or more classes, whereas a `rdfs:range` states that the values of a property are instances of one or more classes. They do not provide any direct way to indicate property restrictions, which are provided by richer Web ontology languages such as OWL.

An example for RDFS can be seen in listing 2.4. The code contains only a description of classes, properties, no individuals, and may be an ontology for the RDF file in listing 2.3.

Listing 2.4: RDFs example.

```
ex1:olympicgames a rdfs:Class .

ex1:summergames
  a rdfs:Class ;
  rdfs:subClassOf ex:olympicgames .

ex1:city a rdfs:Class .

ex1:year
  a rdf:Property ;
  rdfs:domain ex:olympicgames ;
  rdfs:range rdfs:Literal .

ex1:athletes
  a rdf:Property ;
  rdfs:domain ex:olympicgames ;
  rdfs:range rdfs:Literal .

ex1:hostcity
  a rdf:Property ;
  rdfs:domain ex:olympicgames ;
  rdfs:range ex:city .

ex1:name
  a rdf:Property ;
  rdfs:domain ex:city ;
  rdfs:range rdfs:Literal .

ex1:population
  a rdf:Property ;
  rdfs:domain ex:city ;
  rdfs:range rdfs:Literal .
```

Additional information on RDF and RDFS can be found in the W3C recommendations by Cyganiak et al. (2014) and by Brickley et al. (2014).

2.2.4 *Ontology Language*

The OWL is a specification by the W3C. OWL exceeds conventional languages like XML, RDF and RDFS in expressing meaning and semantics. Additionally is it able to create machine interpretable content on the Web (McGuinness & Van Harmelen, 2004).

An OWL ontology consists of classes and properties as known from RDFS. However, in OWL, these classes and properties can be placed in complex relationships with each other. These complex relationships are described by using a set of constructors that come from predicate logic. In OWL, there are two types of properties. Firstly, object properties connecting individuals with individuals. Secondly, data type properties, connecting individuals with data values. In addition, one can use OWL to declare relationships between classes. It is possible to set classes as disjoint or equal. The relationship between individuals is also very important. It can be depicted that different identifiers refer to the same individual (Hitzler et al., 2008).

OWL is part of the W3C recommendations related to the Semantic Web. It has three sublanguages which adapt to different range of application and needs of the ontology developers. The successor for OWL is OWL 2 which is very similar to OWL 1. All OWL 1 ontologies are still valid OWL 2 ontologies. OWL 2 adds new features such as: keys, property chains, richer data-types/data-ranges, qualified cardinality restrictions, asymmetric/reflexive/disjoint, properties and enhanced annotation capabilities (Group, 2012).

The code presented in listing 2.5 is an example of an OWL file handling Olympic Summer Games ontology similar to listing 2.4. OWL brings a new vocabulary to work with classes, data properties, object properties, restrictions, and individuals. Interesting is the `owl:sameAs` property, which allows to link a individual to other individuals.

Listing 2.5: OWL example.

```
<http://www.example.org/olympicgames> rdf:type owl:Ontology .

ex1:olympicgames a owl:Class .

ex1:summergames
  a owl:Class ;
  rdfs:subClassOf ex:olympicgames .

ex1:city a owl:Class .

ex1:year
  a rdf:Property ;
  rdfs:domain ex:olympicgames ;
  rdfs:range xsd:gYear .

ex1:athletes
  a owl:DatatypeProperty ;
  rdfs:domain ex:olympicgames ;
  rdfs:range xsd:integer .

ex1:hostcity
  a owl:ObjectProperty ;
  rdfs:domain ex:olympicgames ;
  rdfs:range ex:city .

ex1:name
  a owl:DatatypeProperty ;
  rdfs:domain ex:city ;
  rdfs:range xsd:string .

ex1:population
  a owl:DatatypeProperty ;
  rdfs:domain ex:city ;
  rdfs:range xsd:integer .

ex1:games2016
  a owl:NamedIndividual, ex:summergames ;
  ex1:year "2016"^^xsd:gYear ;
  ex1:hostcity ex1:rio ;
  ex1:athletes "11237"^^xsd:integer ;

ex1:rio
  a owl:NamedIndividual, ex:city ;
  ex1:name "Rio_de_Janeiro" ;
  ex1:population "6.429.923" ;
  owl:sameAs <http://dbpedia.org/resource/Rio_de_Janeiro>
```

2.2.5 Language Overview

McGuinness and Van Harmelen (2004) described all the languages recommended by the W3C and ordered them by their cardinality in terms of semantic meaning:

XML is a markup language which is readable by humans and machines. It

provides a surface syntax for structured documents but is not able to add semantic meaning to these documents.

RDF is a data model for resources and relations between them. It provides simple semantics for this data model. In general, these data models can be represented amongst others in a XML syntax.

RDFS is a vocabulary for describing properties and classes of RDF resources including semantics for hierarchies of such properties and classes.

OWL adds more vocabulary for describing properties and classes such as: relations between classes, cardinality, equality, richer typing of properties, characteristics of properties, and enumerated classes. OWL comes in three sub-languages.

OWL-Lite has a simple classification hierarchy and restrictions. It allows cardinality constraints but only for values of 0 or 1. Also it has a low formal complexity.

OWL DL has a maximum expressiveness while keeping computational completeness and decidability. It contains all OWL constructs but under certain restrictions of their usage. OWL DL complies the requirements of Description Logics which forms the formal foundation of OWL.

OWL Full brings maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. It allows an ontology to augment the meaning of the pre-defined vocabulary. It is unlikely that any reasoning software will be able to support complete reasoning for every feature of OWL Full.

Overall, it can be said that it is already possible to structure and link information in RDF. However, only with RDFS and OWL there is a possibility to describe information in a more complex and logical way (Hitzler et al., 2008).

2.2.6 Linked Data query language

SPARQL Protocol and RDF Query Language (SPARQL) is a standard for querying information specified in RDF. It is based on simple RDF requests in the form of graph patterns. In addition, SPARQL includes advanced functions for constructing more complex request patterns, use additional filter conditions and for formatting output. Basically, the turtle syntax is used to represent the request graphs. SPARQL also uses query variables that can be used to determine specific elements in the request graph as a

result. Therefore, simple RDF subgraphs can be looked for in the given database. SPARQL makes very weak semantic assumptions that only take the simple RDF inference into account. Neither RDFS nor OWL are directly supported. This means that simple graph patterns with no special semantics in SPARQL play a supporting role (Hitzler et al., 2008).

In general, a SPARQL processor scans every saved triple (graph) according to match the searched “pattern” triple (sub graph). Such a triple can consist of defined values and variables. As result, all the triples that match the search pattern are delivered. Structured, hierarchical search and filtering allows the result to be delivered as desired. An important feature of SPARQL is the possibility of federated queries. Some endpoints have the capability to send queries to other endpoints. That enables querying and combining multiple datasets from different sources. The exact description on federated queries is given by Prud’hommeaux et al. (2013).

GeoSPARQL is a spatial extension to SPARQL. It allows to query geographic information. Therefore, it delivers an SPARQL/OWL vocabulary, a set of SPARQL extension functions and a set of rules for query transformation. GeoSPARQL uses Well Known Text (as defined by Simple Features or ISO 19125) (WKT) serialization to handle geometry data (Perry & Herring, 2012). It is also a Open Geospatial Consortium (OGC) standard and recommendation.

Battle and Kolas (2012) give a more practical review on GeoSPARQL. It is pointed out that GeoSPARQL consists of three main parts:

- Vocabulary to represent features, geometries, and their relationships.
- Spatial functions for use in SPARQL queries.
- Query transformation rules.

The GeoSPARQL ontology is based on the OGC Simple Feature model. It provides the main class `geo:SpatialObject` with two subclasses, `geo:Feature` and `geo:Geometry`. These classes can be combined with an user specified ontology. Features and their geometries are connected via the `geo:hasGeometry` property (Battle & Kolas, 2012). An example for an ontology using GeoSPARQL is given in listing2.6.

Listing 2.6: GeoSPARQL ontology example.

```
ex1:city a owl:Class;  
        rdfs:subClassOf geo:Feature .
```

The listing combines OWL with the GeoSPARQL vocabulary to create an ontology that describes `ex1:city` as a subclass of `geo:Feature`. An

Table 2.1: Topological query functions.

OGC Simple Features	Egenhofer 9-i model	RCC8
geof:sfEquals	geof:ehEquals	geof:rcc8eq
geof:sfDisjoint	geof:ehDisjoint	geof:rcc8dc
geof:sfIntersects	geof:ehMeet	geof:rcc8ec
geof:sfTouches	geof:ehOverlap	geof:rcc8po
geof:sfWithin	geof:ehCovers	geof:rcc8tppi
geof:sfContains	geof:ehCoveredBy	geof:rcc8tpp
geof:sfOverlaps	geof:ehInside	geof:rcc8ntpp
geof:sfCrosses	geof:ehContains	geof:rcc8ntppi

individual of this ontology can be seen in listing 2.7. Since GeoSPARQL is based on the OGC Simple Feature Model, it is possible to use the following geospatial datatypes: Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon.

Listing 2.7: GeoSPARQL individual example.

```
ex:rio a ex:city;
    rdfs:label "Rio_de_Janeiro";
    geo:hasGeometry ex:point1 .
ex:point1 a geo:Point;
    geo:asWKT "POINT(-43.177128,-22.905392)"^^geosf:WKTLiteral .
```

If no coordinate reference system is specified, World Geodetic System 1984 (WGS84) is used by default. An explicitly determined Coordinate Reference System (CRS) can be seen in 2.8.

Listing 2.8: GeoSPARQL literal with CRS information.

```
"<http://www.opengis.net/def/crs/_EPSG/0/4326>_POINT(-43.177128,-22.905392)"^^geosf:WKTLiteral
```

GeoSPARQL makes it possible to query for topological spatial relations and non-topological spatial relations. Topological spatial relations can be used in two ways: applied as a property in the triple pattern or as a filter function. Property relations are used between `geo:Feature` and/or `geo:Geometry` objects while topology functions take the geometry literals as parameters. Both types use the topological functions defined in table 2.1. Non-topological query functions can only be used as filter functions. These operator functions take multiple geometries as predicates and produce new geometries or datatypes as a result (Battle & Kolas, 2012). An listing of the functions is given in table 2.2.

An example for a GeoSPARQL query, using a filter to find every city within a polygon, is given in listing 2.9.

The last functionality of GeoSPARQL deals with query transformation

Table 2.2: Non-topological query functions.

geof:distance
geof:buffer
geof:convexHull
geof:intersection
geof:union
geof:difference
geof:symDifference
geof:envelope
geof:boundary

Listing 2.9: GeoSPARQL query with filter.

```

SELECT ?x
WHERE {
  ?x a ex1:city;
      geo:hasGeometry ?geom .
  FILTER(geof:within( ?geom, "POLYGON((-45.945681_-28.127422, ...))"^^geof:WKTLiteral))
}

```

rules. Here, queries are modified that the `geo:Geometry` object is always taken in account no matter if the `geo:Feature` is used in the initially query. This is achieved by using the `geo:defaultGeometry` property and query rewrite rules. This feature provides a more intuitive approach to geospatial querying (Battle & Kolas, 2012).

As we have seen, GeoSPARQL comes with a lot of features and functions. However, for the user of a spatial enabled triplestore, in most cases, is only the data types and the keywords supported by the implemented triplestore/endpoint of significance, because not all products provide the full capability of GeoSPARQL.

The detailed documentation on GeoSPARQL is given by Perry and Herring (2012).

2.3 PUBLISHING LINKED DATA

The act of publishing Linked Data is very versatile and depends on the use case. The following chapter is all about the steps needed to create Linked Data and publish it on the Web. Also some alternative processes to the common Linked Data approaches will be given. In Bizer et al. (2009), the procedure of publishing Linked Data on the Web is summarized in the

following tasks:

- Choosing URIs.
- Create vocabularies.
- Link generation.
- Metadata.
- Usage of publishing tools.

The mentioned tasks will be explained in detail in the following sections.

2.3.1 Choosing Uniform Resource Identifiers

When thinking about creating and publishing Linked Data, a basic structure has to be respected. This structure is given by four rules set by Tim Berners-Lee (2006):

- Use URIs as names for things.
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI it provides useful information, using the standards (RDF, SPARQL).
- Include links to other URIs so that they can discover more things.

These rules are called “Linked Data principles” and can be seen as guidelines for publishing and connecting data using the infrastructure of the Web without neglecting its architecture and standards (Bizer et al., 2009).

As already mentioned, the URI is one of the key concepts of the Linked Data structure. Every resource is addressed and connected by these identifiers. In the Web of data, there can be endless URIs and entities handling the same real life item. The paper by Bizer et al. (2009) points out that entities are usually generated independently from each other and come from different sources. Diverse URIs that describe the same things are called URI-aliases. These URI-aliases make it possible to have different points of access to an entity. Depending on the type of source and used application, things can develop varying characteristics. The Linked Data approach makes it possible to make all expressions usable under one entity. A simple realization of such URI-aliases can be done either by directly linking the instances or by using the corresponding owl vocabulary in the form of `owl:sameAs` links:

owl:sameAs <<http://dbpedia.org/resource/Graz>>

Therefore, a corresponding namespace has to be chosen at the beginning of the publishing process. All resources and data, which will be generated later, will have URIs according to this namespace. Of course, the URIs have to be dereferencable (retrievable).

2.3.2 Defining Vocabulary

Every dataset in the Web of Data uses vocabularies to define a meaningful ontology. The choice of the vocabulary depends primarily on the type of data that will be published and the Universe of Discourse. The ontology can be hosted on a web server to grant easy access for the users of the dataset. Therefore, the user can explore the ontology to understand the structure and hierarchy of the data. When it comes to the creation of such an ontology, Bizer et al. (2009) recommend the usage of existing vocabulary like Friend of a Friend (FOAF). The FOAF vocabulary was created by the FOAF project which targets to link persons and their social information over the Web. However, in some cases there is the need to create a new ontology.

Case 1

Creating an ontology from scratch is a highly subjective progress. Every person has its own biased view on topics and things and when it comes to describing these things and giving them a meaning, the ontology may vary depending on the person that created it. Noy and McGuinness (2001) point out that there is no “correct” way or methodology for developing ontologies. Nevertheless, following rules are given:

- There is no one correct way to model a domain. There are always viable alternatives. The best solution mostly depends on the application that one has in mind and the extensions that one anticipates.
- Ontology development is necessarily an iterative process.
- Concepts in the ontology should be close to objects (physical or logical) and relationships in your domain of interest. These are most likely to be nouns (objects) or verbs (relationships) in sentences that describe one’s domain.

Furthermore, some possible steps towards creating an ontology are given in this paper. The following points are described in more detail:

Determine the domain and scope of the ontology: In this first step it is considered to answer some simple questions about the domain and the scope of the ontology. These questions would be:

- What is the domain that the ontology will cover?
- For what are we going to use the ontology?
- For what types of questions the information in the ontology should provide answers?
- Who will use and maintain the ontology?

The answers to these questions will be the initial point in this iterative process.

Consider reusing existing ontologies: In the second step it is shown that it is sometimes useful to use existing ontologies someone created before. This is a concern if the application to be created should interact with other applications. There are a lot of ontologies available on the Web that can easily be imported into a project.

Enumerate important terms in the ontology: This is simply a matter of listing certain keywords that are to play a role in the ontology. These terms will be structured and divided into classes, subclasses, properties and subproperties later.

Define the classes and the class hierarchy: The paper presents three possible ways to create a class hierarchy. A top-down approach, a bottom-up approach, and a combination of both. The top-down approach starts with the definition of the most general classes in the domain and subsequent specialization of these. On the other hand, the bottom up approach starts with the definition of the most specific classes.

Define the properties of classes—slots: After the classes have been defined, special object properties can be assigned to them. These object properties are attached to the classes and their individuals and serve their description.

Define the facets of the slots: This step is all about defining domain, range, datatypes and cardinalities for the individuals. Here, the domain defines what class a property describes, while the range defines what classes are allowed for the property to interact with. The cardinality, on the other hand, defines how many values a property can have. Datatypes are attributes of data properties. These properties will generate individuals with a specific datatype like integer or sting.

Create instances: When all previous steps are finished, individual instances of the ontology can be created.

All in all, it can be said that the creation of an ontology is a highly fluid and iterative process that needs a clear definition of its domain and application.

Case 2

Deriving an ontology from an existing domain of discourse is another approach for creating a new ontology. Using data from a relational database could be one of these cases. Having data stored in a relational database will be a common origin in developing Semantic Web applications and serving Linked Data. There are a lot of software-tools available which can create ontologies on the fly using the existing table structure in a database-schema. However, to get a better understanding of the data and the creation of its ontology, it can be also generated manually. Here, the complex design of the relational database, focused on logical data structure, could make it difficult for humans to understand the semantics of the data and create a meaningful ontology out of it. So the first step might be to transform it back to an entity-relationship model, if it is not available anyhow.

An ERM is a useful tool for handling data regarding its semantics. It follows the attempt that information in the real world consists of entities and relationships. Things in the real world refer to entities and the association between these are called relationships. The Entity Relationship Model (ERM) uses the semantic information to manage functional dependencies of data and create a database model. It is also written that the ERM can be used as a basis for an unified view of data (Chen, 1976).

The usage of the semantics in the ERM is a good starting point for creating an ontology. In addition, the model has further properties which can directly be mapped. Myroshnichenko and Murphy (2009) did some research in finding and describing rules for mapping ERMs to an OWL Lite ontology. Therefore, five mapping rules are explained. These rules are based on the approach to map well-formed ERMs into Object Data Management Group class definitions by Elmasri and Navathe (1994). The rules can also be interpreted in tabular form which is shown in table 2.3.

Table 2.3: Correspondence between components of well-formed ERMs and OWL Lite ontologies according to Myroshnichenko and Murphy (2009).

ER-Schema	Ontology
Entity	Class
Strong Entity	Class
Weak Entity	Subclass of the according strong entity
Attribute	Datatype property
Single-valued Attribute (nullable)	Functional datatype property
Single-valued Attribute (not nullable)	Functional datatype property with min constraint set to one
Multi-valued Attribute (nullable)	Datatype property
Multi-valued Attribute (not nullable)	Datatype property with minimum constraint set to one
Key Attribute	Functional datatype property with minimum constraint set to one
Composite Attribute	Class with properties corresponding to components of the composite attribute
Binary Relationship without Attributes	Pair of inverse object properties
Binary Relationship with Attributes	Class with datatype properties corresponding to the relationship's attributes and two pairs of inverse object properties associating participating entity classes and the relationship class
Ternary Relationship	Class with three pairs of inverse object properties associating the participating entity classes and the relationship class
Participating Entity Role Name	Name of the appropriate object property in the pair of inverse object properties manifesting a relationship without attributes
Min Cardinality	Min cardinality restriction
Max Cardinality	Max cardinality restriction

2.3.3 Link Generation

In this step it is necessary to enhance the original data with Linked Data information. Research has to be done so that entities of the original data

can be linked to resources hosted at an other domain. These links must be added to the data that will then be converted to Linked Data including RDF links.

According to Bizer et al. (2009), this step should include the usage of automated or semi-automated approaches. Furthermore, it is shown that various RDF link generation frameworks are available, that provide declarative languages for specifying which types of RDF links should be created, which combination of similarity metrics should be used to compare entities and how similarity scores for specific properties are aggregated into an overall score.

Due to the complexity of some of these frameworks, a more easy approach could be to create a program that crawls the Application Programming Interface (API) or a data dump from a provider of Linked Data. Big suppliers of Linked Data are DBpedia or GeoNames⁶. Due to the fact that data crawling by using the API generates lot of traffic in form of requests for the data provider, it is recommended to download a dump file from the servers and search this files locally when dealing with greater tasks.

Before searching for appropriate Linked Data, it has to be ensured that the original data has a well-formed structure. Tools like OpenRefine⁷ provide such functionality. Errors according to text-encoding, data values or date and time formats can be fixed with such a method. This instance is significant for the final step of publishing Linked Data.

2.3.4 Metadata

Hartig (2009) recommends to publish metadata along with Linked Data. Metadata makes it possible for the user to detect the provenance of the Linked Data. This allows the user to estimate the quality of the data and the suitability for their purposes. Techniques which provide this information are “Dublin Core” terms or the “Semantic Web Publishing Vocabulary” (Carroll, Bizer, Hayes, & Stickler, 2005).

2.3.5 Publishing Tools

Ontology Editor: An ontology editor is a tool which can help to create RDF vocabularies. These editors make it possible for non-experienced users to create meaningful ontologies. They take away the necessity

⁶<http://www.geonames.org/>

⁷<http://openrefine.org/>

to handle with the file structure and the syntax. A famous ontology editor is Protégé⁸. This tool is widely used and often recommended to novices to Linked Data. Protégé is an open-source software that uses the current W3C standards. Beneath the ability to include lots of plug-ins, it comes with an included visualization tool and a reasoner. A semantic reasoner applies multiple reasoning tasks on the ontology and searches for errors that occur by doing so.

Triplestore: The simplest way to store triples is by hosting the RDF files. This can be a good solution for small datasets. However when it comes to host larger datasets, a triplestore is the way to go. A triplestore is a kind of database that stores OWL or RDF triples. The stored triples can be received (read) and manipulated (updated). Triplestores come in different system architectures. Some of them use existing database management systems such as PostgreSQL⁹ or MySQL¹⁰ as base, while others are build from scratch. For querying the stored data, the W3C standard SPARQL is used. GeoSPARQL is the extension to SPARQL when it comes to handle spatial data and a OGC standard. Not all triplestores have the ability to store and query spatial data. Some of the spatial triplestores are Parliament¹¹, Strabon¹², Virtuoso¹³ Stardog¹⁴ and GraphDB¹⁵.

Virtual triplestore: In a virtual triplestore, data is stored in relational database and mapped in real time to be served as triples. This is a great concept for publishing Linked Data that is already hosted in a relational database management system. In order to achieve this, a mapping language is used. In simpler words: relations (tables) will be mapped to RDF classes and foreign keys or values to properties. For the user, a virtual triplestore offers the same experience as a native triplestore. An example for a virtual triplestore is the “D2RQ”¹⁶-Platform that consists of two parts: the “D2R Server” that offers data access via HTML, RDF-browsing and SPARQL querying and the “D2RQ Engine” which processes the mapping to the relational database. D2RQ does not support spatial data.

SPARQL Endpoint: A SPARQL endpoint is a service that allows the user to interact with the query engine of a triplestore. This service is

⁸<http://protege.stanford.edu/>

⁹<https://www.postgresql.org/>

¹⁰<https://www.mysql.com/>

¹¹<http://parliament.semWebcentral.org/>

¹²<http://www.strabon.di.uoa.gr/>

¹³<https://virtuoso.openlinksw.com/>

¹⁴<http://www.stardog.com/>

¹⁵<http://graphdb.ontotext.com/>

¹⁶<http://d2rq.org/>

often provided as web application. Most triplestores have an included SPARQL endpoint. Modern endpoints have the ability to perform federated queries. This allows the user to query multiple endpoints at a time and create comprehensive queries.

Linked Data interface: According to Bizer et al. (2009), Linked Data should be provided as dereferenced links. Therefore, a Linked Data Interface has to be implemented. With such an interface, human users will be presented a HTML web page and a tabular representation of the data when accessing specific RDF resource, while a RDF browser, RDF crawler or software agent, however, will be given the pure RDF data. Nearly all triplestores come with a Linked Data interface. For those who do not carry it, there is a software called Pubby¹⁷. This software adds a Linked Data interface functionality to a triplestore by using SPARQL DESCRIBE queries.

2.3.6 Additional Approach

An additional approach to publish Linked Data is creating a wrapper for a API. Nowadays, many Websites provide a API to expose their data to a broader audience. The information generated this way is represented in formats like XML, JavaScript Object Notation (JSON) or geoJSON. Linked Data wrappers assign HTTP URIs to the resources that are provided by the API. With this approach and an existing API, it easy to expose new sources for Linked Data (Heath & Bizer, 2011). The downside of these practice maybe the native absence of a SPARQL query functionality and a Linked Data interface. All in all, this is a quick but custom solution.

2.4 LINKED OPEN DATA

The principle of Linked Open Data came up with the Linked Open Data Project. Aim of the project was creating a Web of Data by finding existing open data sources, converting the data to RDF and publishing it as Linked Data. Some datasets such as DBpedia or GeoNames have become hubs in this Linked Open Data cloud. These datasets provide URIs and RDF descriptions for nearly every thing or place of the real world and, therefore, more specialized and smaller datasets are often linked with these (Bizer et al., 2009).

In Auer et al. (2007), DBpedia is called the nucleus for the Web of Open

¹⁷<http://wifo5-03.informatik.uni-mannheim.de/pubby/>

Data. This means that everything is linked to this origin and every exploration can start from here. Therefore, various datasets have to interlinked with DBpedia data through RDF links.

2.5 LINGUISTICS AND SPATIOTEMPORAL LINKED DATA

How does linguistics and its branch dialectology fit in the presented spatiotemporal Linked Data approach? The answer to this question is: very well. According to Scholz et al. (2016), language and dialects associated with areas have fuzzy or crisp borders. As language changes over time, these borders are changing. In more simplified terms, language regions can be seen as polygons thus borders are time-dependent. This dependency on time goes back to phenomena called language death, language revitalization and language shift. These phenomena also apply to dialects as its specific terms change, arise or vanish even faster due to fact that dialect words are mostly spoken and rarely written.

CHAPTER 3

APPROACH

In order to answer the scientific questions, an experiment is done. The experiment is realized by a proof-of-concept prototype which will show the capabilities of the approach in a test environment. In this chapter, the chosen approach will be presented in detail. The approach is strongly inspired by the one given by Bizer et al. (2009) and described in section 2.3. It is well suited for publishing existing data as Linked Data and, therefore, fits exactly the needs of this project. This chapter will give some additional information about why a specific step is chosen and how the prototype will be realized. A graphical interpretation of the approach is given in figure 3.1.

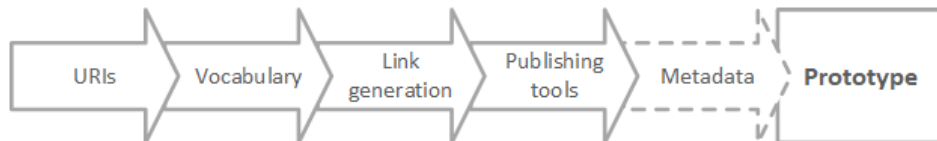


Figure 3.1: Diagram of the approach.

It shows that the approach is structurally based and aims at producing a prototype. The individual steps are shown as arrows, because the next step always requires the previous one. The only exception is the step of metadata creation which is optional but improves the data quality substantially. The technical structure of the experiment can be found in chapter 4.

3.1 BASIC CONDITIONS

The approach has some basic conditions to which it tries to hold on, while developing the prototype. These restrictions should guarantee a high grade of interoperability and customization. The basic conditions can be summed up to:

- Use World Wide Web Consortium (<http://www.w3.org/>) (W3C) and Open Geospatial Consortium (OGC) recommendations and standards.
- Use open source software.

- Use software that has an active community.

The argumentation for this points is given in the following subsections.

3.1.1 Software

Bitzer and Schröder (2005) defined commercial software as a software which, through certain licensing conditions, prohibits or limits usage, modification and duplication. Modification is often not an option as the source code is not available. The possibilities for usage are limited as the source code is replaced by a compiled code. It is further pointed out that in contrast to commercial software, the source code of open source software is freely available. The basic idea and main aim of open source is free usage and the possibility for further development by the user. The user is free to duplicate, modify, and distribute open source software. The possibilities for modification and the advantage of active user communities are the reasons that led to the use of open source software primarily for this prototype. Therefore, the prototype can be used and modified in the way it is required, without the need to worry about licensing.

3.1.2 Standards

On the web of documents, web standards are defined by the W3C and its working groups, hence, they try to enhance the interoperability of web-based projects. Similar standards can be found for the Semantic Web. Burgos (2011) showed that there are many well-known standardized technologies such as Resource Description Framework (RDF), OWL Web Ontology Language (OWL) and SPARQL Protocol and RDF Query Language (SPARQL). These standards ensure simple data exchange in the means of syntax and semantic. As it takes some time for a technique to become a standard, these recommendations sometimes are not the cutting-edge technology. However, the advantage here is that the standards are used by many people in the Semantic Web community and are well tested and documented. According to the Linked Data principles given by Tim Berners-Lee (2006), it is defined in rule three to use the standards. Standards and recommendations regarding the Semantic Web are given by the W3C. Standards and recommendations regarding spatial technologies are given by the OGC.

3.2 LINKED DATA PUBLISHING

3.2.1 *Resource identification*

Since this is about building a prototype, that not necessarily has to be online, the choice for a namespace and significant Uniform Resource Identifiers (URIs) is not very important. According to this condition, some local-host URIs can be used to please this task. For a later usage of the Linked Data storage system, using specific Hypertext Transfer Protocol (HTTP) domains and URIs must be considered.

3.2.2 *Vocabulary creation*

Two possible cases of creating an ontology are given in subsection 2.3.2. In the first case, the ontology is created from scratch. Here, the triples can not be transformed directly and some manual operation, respectively some scripting is needed to generate triples. The main advantage is that the ontology can be build just as it is needed. The second case, that is introduced, can be used if data, stored in a relational database, is available and it should be the basis to create the ontology and further on to generate triples. The advantage of this approach is simplicity and speed. Triples and their ontology can be created partially automated. The disadvantage is that we are bounded on the relational schema. That is why the ontology can not be adjusted at will. Because this project is based on relational database and its Entity Relationship Model (ERM), the second approach is a good choice to create a vocabulary that fits the used data and it will be sufficient for testing purposes.

3.2.3 *Link generation*

As already mentioned, there is plenty of open data in the Web of Data. For this project it is considered to link the lexical dataset of the “Datenbank der bairischen Mundarten in Österreich electronically mapped” (dboe@ema) project to DBpedia and GeoNames datasets. DBpedia should be chosen because Auer et al. (2007) defined it as a nucleus of the Semantic Web. A DBpedia dataset is interlinked with various other data sources on the Web and allows DBpedia users to discover further information. It is also mentioned that DBpedia, with its broad topic coverage, intersects with many datasets and, therefore, makes an excellent “linking hub”. Araujo, Houben, Schwabe, and Hidders (2010) also defined GeoNames as an hub to which

an increasing number of data sets are connected. It provides RDF descriptions of millions of geographical locations worldwide. This definition makes it clear that GeoNames is an important instance of Geographic Information Science (GIScience) in the Linked Open Data (LOD) cloud and is almost indispensable to link to when creating a new Linked Data source with spatial background. The actual generation of the Links is achieved by using python scripts in combination with the Application Programming Interface (API) from Wikipedia and GeoNames. The detailed workflow is given in the subsection 4.4.2

3.2.4 Usage of Publishing Tools

Spatial approach

As already mentioned, the standard to handle geospatial Linked Data is GeoSPARQL. This allows us to use spatial data saved in a triplestore just like data stored in a spatial relational database such as “PostGIS”. However, not every triplestore is capable of processing geospatial data. So what happens if one wants to extend data with spatial information but does not want to migrate all of it to a new triplestore? If a dataset is stored in the non-spatial triplestore “A”, resources can easily be linked to spatial triples in the spatial triplestore “B”. If each of these triplestores has a SPARQL endpoint, federated queries can be made allowing comprehensive tasks to be fulfilled. The original data has to be extended with statements linked to their spatial counterpart. In figure 3.2, an example for this approach is given.



Figure 3.2: A graph representing a location with its spatial information stored on a separate source.

The reason why the split data approach is preferred rather than the ap-

proach of dumping the complete database into an RDF file and store it in an spatial triplestore, is in the maintenance. The spatial information in the `dboe@ema` is in a closed state, while the lexical datasets are incomplete and new datasets get added from time to time. If a single triplestore is used for all the data, every time a new dataset is added to the relational database, it has to be dumped to an RDF file and transferred to the triplestore. However, with the here presented approach, the new dataset from the database automatically is mapped to RDF triples by the D2R Server. The spatial data only needs to be transferred to the spatial triplestore once since it remains constant.

Temporal approach

In Linked Data, time can be represented as part of the primitive data types of the XML Schema. All triplestores can filter for time using logic operators. Time is mostly expressed by using the data type “`dateTime`”. This data type only allows discrete points in time to be stored. However, what if it is needed to store large time periods? Here, the standard vocabulary has to be enhanced. A way to achieve this is by using the OWL-time ontology. This ontology is build to represent time and has the possibilities to handle a lot of time depending additional information like timezones, day of the week, day of the year and so on. A more lightweight solution is to declare instances for a start-time and an end-time and use a standard XML Schema data type like “`dateTime`”. A graph handling a time period is given in figure 3.3.

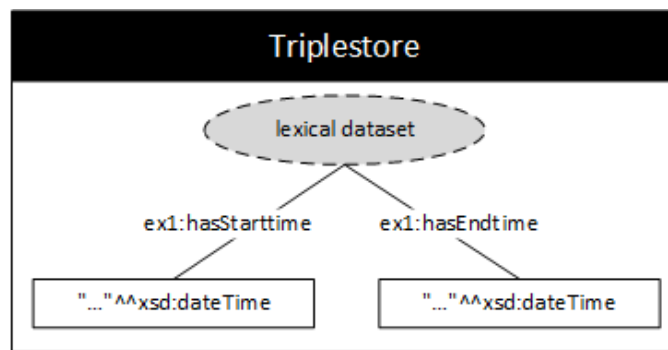


Figure 3.3: Example for a graph handling a time period.

3.2.5 Metadata

Hence the data of the `dboe@ema` project is considered an proof of concept data set, it is under further development and construction. Thereby the

Metadata is dealt with within the current project “exploreAT!”. For the Prototype the basic metadata is automatically created by D2RQ publishing tool. If more detailed information on the data is needed, it can easily be added afterwards with D2RQ by using so called metadata templates.

3.3 LINKED DATA EXPLORATION

3.3.1 Queries

The introduced approach for the publication of Linked Data has a small disadvantage when it comes to queries. Since the data is stored virtually in two different triplestores, each with its own (Geo)SPARQL endpoint, the complete data can not be treated with a standard (Geo)SPARQL query. The solution to this problem is the so-called federated query. As written by Kuhn et al. (2014), federated queries allow for accessing data from different SPARQL endpoints and further to combine the results from multiple sources. A corresponding visualization of the query approach can be seen in figure 3.4.

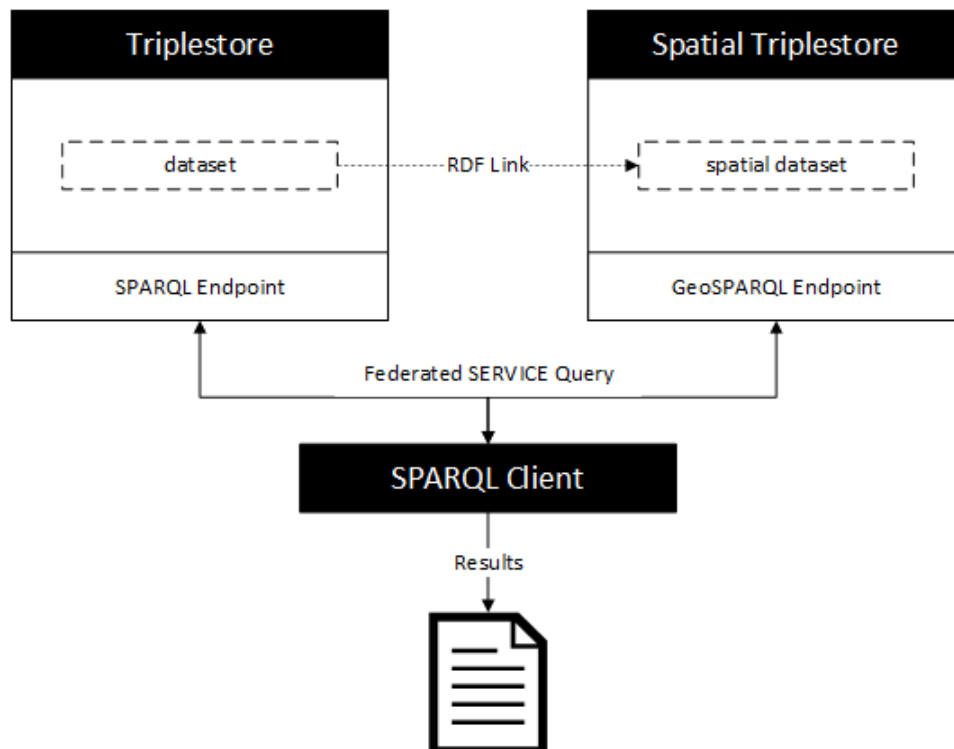


Figure 3.4: Query approach.

It shows a regular triplestore and a spatial triplestore, each with its dedicated (Geo)SPARQL Endpoint. The endpoint is the interface that receives the incoming queries, processes them and returns the results. In order to be able to make these queries, the data must have a common reference. This is created by an RDF link that links the lexical data of the dboe with the spatial data of the spatial triplestore. Thus, the federated queries of the client can be processed, and the results can be interpreted appropriately.

3.3.2 Testing Environment

In order to answer the scientific questions, a testing environment has to be created. The system needs to be tested for its spatiotemporal capability and its benefits compared to the classic database system. Following tasks should be performed:

- Temporal queries,
- Spatial queries,
- Visualization of geodata,
- Data exploration with RDF and Hypertext Markup Language (HTML) browser.

Test applications have to be created for accomplishing these tasks. These applications use Linked Data from the prototype.

CHAPTER 4

PROTOTYPING

In this chapter, the base data of the project and a detailed workflow will be described. Additionally an overview about the used software, its configurations and its usage will be given.

4.1 INITIAL SITUATION

The aim of this project is to create a prototype for the storage of lexical data by using Linked Data technologies. A physical prototype can be used to answer the questions about the capabilities of a Linked Data approach and how it will compete against the classic relational database approach. The thesis is basically a successor to the “dboe@ema¹” project and will proceed in the current project exploreAT! (exploring austria’s culture through the language glass). Therefore, it is a logical approach to use the existing resources as best as possible. According to Bartelme et al. (2016), the project “Datenbank der bairischen Mundarten in Österreich elektronically mapped” (dboe@ema) has the aim of making the extensive collection of dialect data of the Austrian Academy of Sciences (ÖAW) accessible to a broad public. The data used for the project is related to physical paper slips that contain information about a dialect word. Before the project, since 1993, a digitization of the slips has been carried out by using the software package “TUebinger System von TExtverarbeitungs-Programmen” (TUSTEP). The data generated in this way is optimized to the needs of scientific processing of text files but not directly suitable for storage in relational databases. During the project, in a semi-automatic process, the data of the TUSTEP system is migrated into a database structure and connected to spatial data. The created spatial database system is the basis for a web-based GIS. The Data concerning 32000 vouchers about plants and mushrooms was the first one to be published for free, interactive usage. Explanatory reference materials include a bibliography, a register of persons, an index of keywords and a register of places, municipalities and regions were spatial referenced and published.

¹<https://wboe.oeaw.ac.at>

4.1.1 Initial Data

The dboe@ema base material is written on vouchers. These are physically paper slips which contain information about a lemma and its source. Every source has attributes such as a title, a time when it was published or rather when it was created and a place where it belongs to. An example for a paper slip, describing a bottle crate is shown in figure 4.1. The basis for the

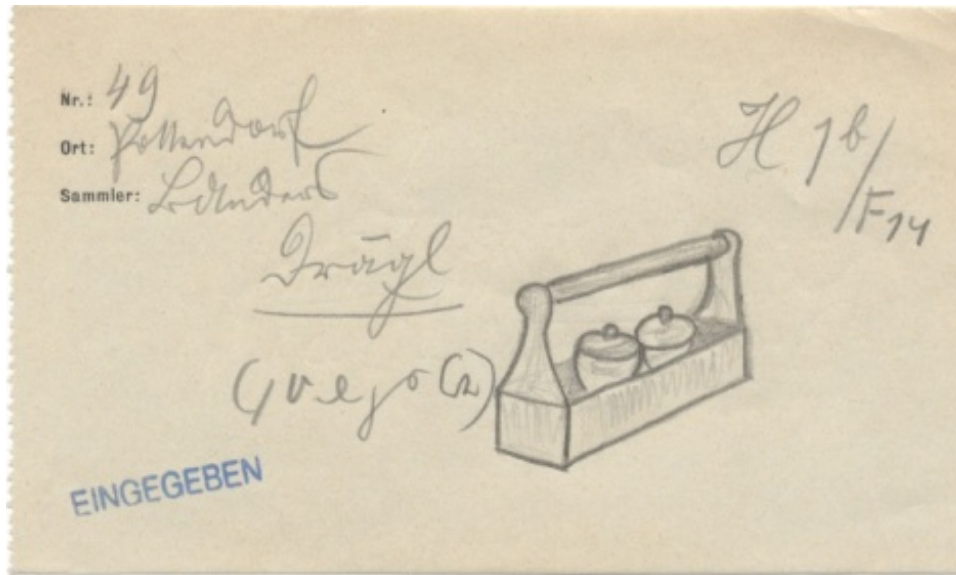


Figure 4.1: A voucher describing a bottle crate.

database model is formed by the Entity Relationship Model (ERM) which is a graphically interpretation of the system and can be seen in a simplified form in appendix A. The relational database and the ERM are the starting points for the subsequent tasks of this thesis. In the center of the data, there is the table that contains the records of the paper slips (“belegzettel_beleg”). This table is essential, because it connects the paper slips (“belegzettel”) to lemmas (“lemma”), meanings (“belegzettel_beleg_bedeutung”) and additional information about the recorded paper slips. A voucher can be a stand alone object or have a source (“quelle”) such as a book or a questionnaire (“fragebogen”). This depends on how the information was retrieved by the collector. In the dboe@ema, spatial information is given in the form of place (“Ort”), municipality (“Gemeinde”) and region (“region”). Each of these entities is stored in a table and has an additional table which contains the geometric information in form of OGC simple features. The names of the locations are given in a short and a long version. Additionally, the hierarchical structure of the locations is considered. Therefore, there are places in a community, communities in a region and regions in superior region. This

structure is represented by 1:n relationships and the usage of foreign keys. A visualization of the tables containing the spatial information is given in figure 4.2. A more detailed explanation is given in section 4.4 The tempo-

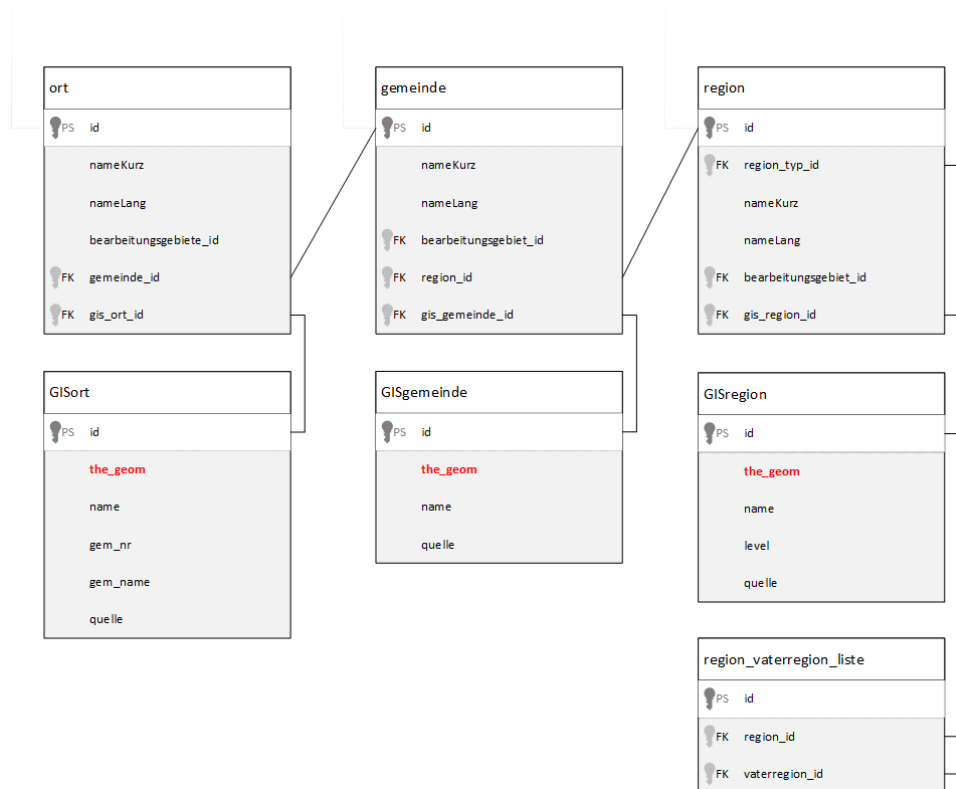


Figure 4.2: Spatial tables in the dboe@ema.

ral information is contained in the tables for the paper slips (“Belegzettel”), source (“Quelle”), Person (“Person”) and the questionnaires (“Fragebogen”). For example, in the voucher (“belegzettel”) table there is an attribute “belegjahr” which marks the year in which the paper slip was created. A person has a day of birth and a day of death. Questionnaires and literary sources have a year of publication. Since the sources were often books, they can have additional time related attributes like the year of the first edition, the publication period or the period in which it could be obtained. The problem with these time values is that they have no proper data type in the dboe@ema and are stored as strongly varying strings. The correct temporal data type assignment is discussed in chapter 4.4. The tables with temporal information are given in figure 4.3. Temporal attributes are written in bold and red. The relational database of the dboe@ema contains about 80 tables. Only 24 of them are used to create this prototype. The reason for this constraint is that some these tables are not used yet, but, first and foremost,

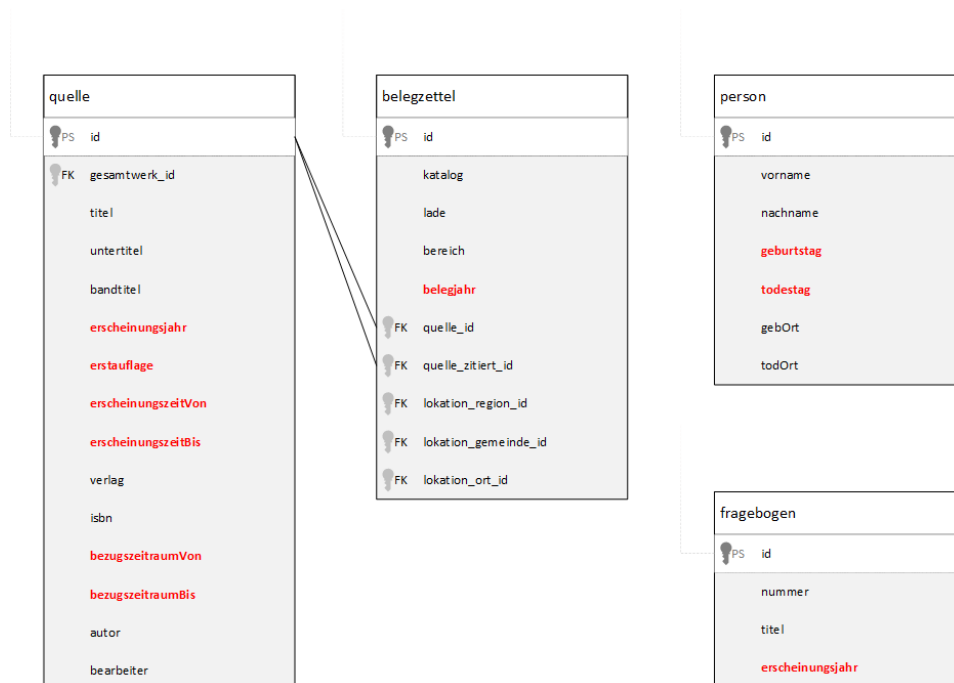


Figure 4.3: Tables with temporal information.

to make this prototype simpler and more straightforward. This is why only the essential tables are considered.

4.1.2 Implementation

When it comes to the implementation of this prototype, the in section 2.3 introduced tasks are used. As already mentioned the goal is to create a working prototype, to store and query lexical information in form of Linked Data. Hence not every task can be fulfilled exactly like it would be when creating an actual implementation for the public usage. Also keep in mind, that this implementation is designed for the `dboe@ema` project and may not fit with other projects due to a different starting position and requirements. The implementation can be seen as a basis for further or other projects and gives a preview about the possibilities of Linked Open Data (LOD) in digital humanities.

During this project, multiple experimental configurations are used and in the end the workflow shown in figure 4.4 is the best way to meet the requirements. The individual steps of the workflow are discussed in detail in the following sections.

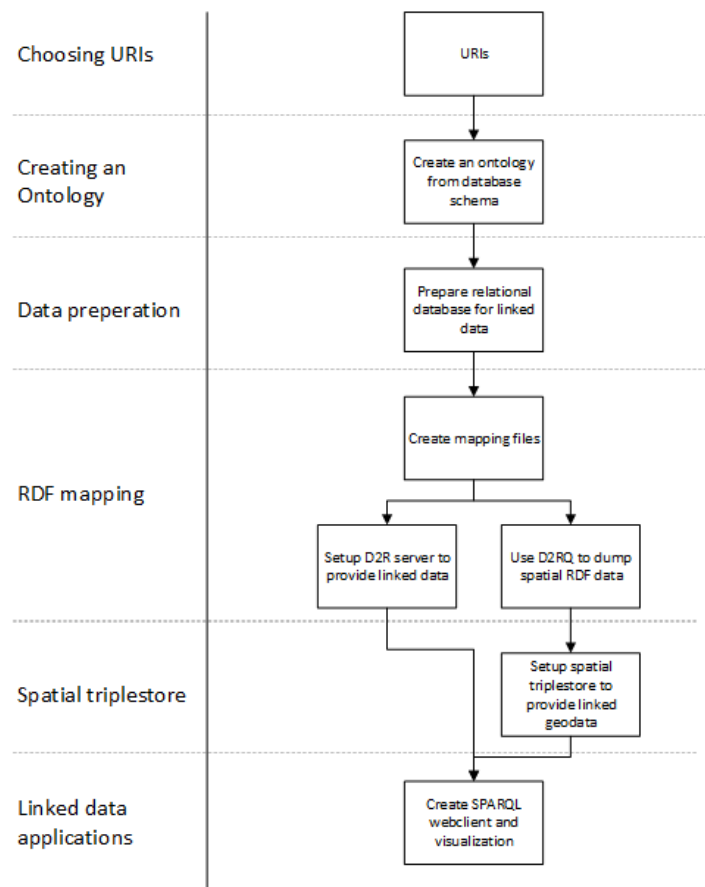


Figure 4.4: Workflow.

4.2 CHOOSING UNIFORM RESOURCE IDENTIFIERS (URIs)

If used on a local machine, every URI is unique and points to a specific resource. Following URI points to the resource of the place “Graz” located on the local server:

```
http://localhost:2020/resource/ort/19298
```

The same resource on a public server may have a URI such as:

```
http://fphotpc66.tugraz.at:8081/resource/ort/19298
```

Here, a Persistent Uniform Resource Locator (PURL) can also be used. According to Berrueta, Phipps, Miles, Baker, and Swick (2008) persistent Uniform Resource Locator (URL)s are URIs from the `http://purl.org/` URI space. These special URIs point to the PURL server which throws the response code 303 (“see other”) to link to the variable URI of the resource. This step is only a possible extension and is hardly used nowadays.

4.3 FROM A RELATIONAL DATABASE TO AN ONTOLOGY

This section is all about creating a meaningful ontology for the lexical dboe@ema data. For transforming relationships to an ontology a ERM is needed. A simplified version of the ERM in Unified Modeling Language (UML) notation is given in appendix A. This graphic shows the structure of the database, consisting of the different tables and attributes and keys. The table in the center of the database handles all digitalized vouchers. Starting from this table, attributes and tables dealing with lemmas, sources and locations are connected. This model gives a good overview about the database, but for creating an ontology, the Chen-Notation is more descriptive. An extract of the dboe@ema data transfered to the Chen-Notation is given in 4.5. In this diagram, entities are shown as boxes, relationships are shaped as diamonds and the attributes have an oval appearance.

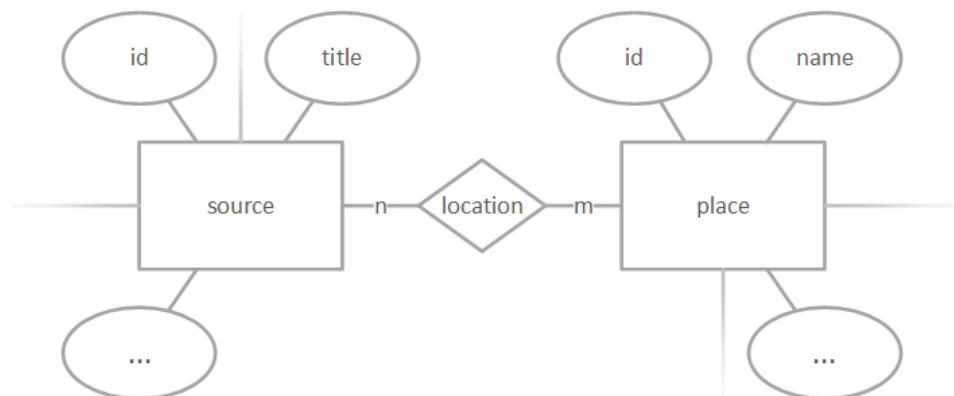


Figure 4.5: A part of the dboe@ema ER-Diagram in Chen-Notation.

Utilizing the mentioned mapping rules we get following results. As shown in listing 4.1 the strong entities “source” and “place” can each be mapped to a class. Note that a fictional URI named “myuri” is used for the following examples.

Listing 4.1: OWL Web Ontology Language (OWL) classes.

```
<owl:Class rdf:about="http://myuri/dboe#source"/>
<owl:Class rdf:about="http://myuri/dboe#place"/>
```

The relationship “location” is transfered to an object property. It is important to refer to the correct domain and range. Each relationship has an inverse property where domain and range is inverted. The names of the properties are adjusted to be more suitable. For example, the relationship “location” are translated to the object property `hasLocation` with its domain in the class `source` and the range in the class `place`. The inverse is `isLocationTo`, with swapped domain and range. Object properties can

have multiple domains and ranges. Listing 4.2 shows the property that a source has a place as location. Noticeable is also the inverse property saying that a place can be a location to a source.

Listing 4.2: Object properties.

```
<owl:ObjectProperty rdf:about="http://myuri/dboe#hasLocation">
  <owl:inverseOf rdf:resource="http://myuri/dboe#isLocationTo"/>
  <rdfs:domain rdf:resource="http://myuri/dboe#source"/>
  <rdfs:range rdf:resource="http://myuri/dboe#place"/>
</owl:ObjectProperty>

<owl:ObjectProperty rdf:about="http://myuri/dboe#isLocationTo">
  <owl:inverseOf rdf:resource="http://myuri/dboe#hasLocation"/>
  <rdfs:domain rdf:resource="http://myuri/dboe#place"/>
  <rdfs:range rdf:resource="http://myuri/dboe#source"/>
</owl:ObjectProperty>
```

Attributes are mapped to data properties. Data properties can also have multiple domains. In the given example, every entity has an attribute called `id` which is an integer number. An example for the created data type properties is given in listing 4.3.

Listing 4.3: Datatype properties.

```
<owl:DatatypeProperty rdf:about="http://myuri/dboe#id">
  <rdfs:domain rdf:resource="http://myuri/dboe#source"/>
  <rdfs:domain rdf:resource="http://myuri/dboe#place"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
</owl:DatatypeProperty>
```

Ontologies can also be displayed as graph, which is more intuitive to understand for humans. A tool for creating ontologies is Protégé. This tool will be discussed in subsection 4.6.2.

4.3.1 *Providing the Ontology*

For users of the Linked Data it may be interesting to know the exact structure of the data. This information can be handy when it comes to applications that work with the provided data, using the vocabulary for other datasets or extending the vocabulary for a different application. In order to provide this information the ontology needs to be published and has to be accessible to people or machines. These two different types of clients have different requirements on the structure of the ontology. A human person that wants to explore the structure of an ontology uses a simple web-browser to navigate the web and process information that is provided in HTML. However, a software can use the raw Resource Description Framework (RDF)/OWL data to gather information. In most cases, even for human users it is enough to publish the RDF/OWL file. These files can then be downloaded and handled with special RDF-editors such as Protégé. To

accomplish this, the ontology is hosted on a local web-server. Therefore, a server software needs to be installed and configured. When using an Apache-based web server also a “htaccess” file must be created. This file includes the rules to rewrite the incoming URL requests and connect them to the ontology file. If a client tries to go to a property or a class of the ontology by using its URI, it automatically gets redirected to the ontology file. As file format turtle is used. The actual content of the `.htaccess` file is given in listing 4.4

Listing 4.4: Apache server htaccess file.

```
# Rewrite engine setup
RewriteEngine On
RewriteBase /ontology

# Rewrite rule to serve \ac{RDF}/XML content from the namespace \ac{URI}
RewriteRule ^dboe$ dboe.ttl
```

4.4 DATA PREPARATION

4.4.1 *Data quality*

In order to use data stored in a relational database, as Linked Data, it has to fulfill some degree of consistency and homogeneity. When data gets recorded by human beings or gets copied from different sources, errors are made. These errors can be cryptically character strings, leading and tailing whitespace, wrong data types or the usage of a wrong syntax for a specified data type. This is particularly evident when it comes to search data or interlink data. The following subsections will give a hint about what inconsistencies occurred and how to dissolve them.

A tool such as OpenRefine² can be used to prepare the data. The purpose of this tool is to fix, clean up and transform messy data. A huge advantage of this program is its ability to sort, filter and group data. Therefore, errors can be seen and patched easily. Figure 4.6 gives an example for inconsistency and wrong characters for describing the year of release of some paper slips and how its displayed in OpenRefine.

Wrong Characters

Wrong characters can have two origins. They can simply be typing errors or encoding errors. Many languages use special characters beneath

²<http://openrefine.org/>



Figure 4.6: Faceted variation of data describing the year of publication.

the regular letters of the alphabet. In German, these addition characters are “ä, Ä, ö, Ö, ü, Ü, ß”. Sometimes, these characters are not supported well by some text encoding standards and so, when text gets copied, it can result in encoding errors. These errors show in form of cryptic character strings, where language-specific characters get replaced by a series of symbols such as “% , & , \$, ? , !”

Unwanted characters can easily be replaced or removed by searching and replacing or removing them. This can be done in OpenRefine by using following command:

```
value.replace('?', '')
```

This simple command replaces the character “?” with no character, and, therefore, removes all “?”.

White spaces

White spaces may not look like a deal-breaker but they can cause problems when later on the data is used later on to find counterparts in the web of linked-data by using string-matching algorithms. In order to get rid of leading- or trailing white spaces in OpenRefine, the data can be sliced or just be extracted with the wanted amount of digits. The following code gets the last four digits and ignores the leading characters or white spaces:

```
value.slice(-4)
```

Another way is extracting a matching string according to “regular expression” syntax:

```
value.match(/(\d{4})/)[0]
```

Data types

If the used data should be able to get queried in a later application, it must be ensured that the correct data types are used. As already seen in the example in figure 4.6, there are many ways to describe the year of publication of a voucher. However, if somebody tries to query this data and finds or filters data over specific years, it will not work under these circumstances. Here, the year has to be an integer number so that the user and the application know what they are dealing with and can work with it.

Due to their complexity and strong variation, most of this wrong datasets have to be replaced manually one by each other.

4.4.2 Linked Open Data

After the process of refining the data is completed, it can be tried to find matching datasets in the web of LOD. In order to link data, we have link to the URL of a specific resource. As already mentioned, we can link individuals by using the owl property `owl:sameAs`. So for example, if one wants to link the individual of the city Graz to the individual of Graz according to the GeoNames data, one has to find the resource within the GeoNames data and create a corresponding data property. In the `dboe@ema` ontology the place “Graz” is represented by the resource `localhost:2020/resource/ort/19298`. At the GeoNames server the place Graz is represented by `http:sws.geonames.org/2778067/`. A triple that links these individuals might be:

```
PREFIX db: <http://localhost:2020/ressource>
<db:ort/19298> <owl:sameAs> <http://sws.geonames.org/2778067/>
```

GeoNames

According to their homepage ³, GeoNames is a database that contains over 10 million geographical names of locations and consists of over 9 million unique features where of 2.8 million populated places and 5.5 million alternate names. In order find out what attributes can be used to link the `dboe@ema` spatial information to GeoNames data, the structure of both data sets need to be compared. Within the `dboe@ema` data spatial information can be found for vouchers, sources and people. This spatial information is divided into:

³<http://www.geonames.org/about.html>

- place (“Ort”)
- municipality (“Gemeinde”)
- region (“Region”)

On an administrative level, places are the smallest unit represented by point data followed by municipalities which are in administration of places and represented by polygons. Figure 4.7 shows the structure of the spatial data and its hierarchy in the dboe@ema.

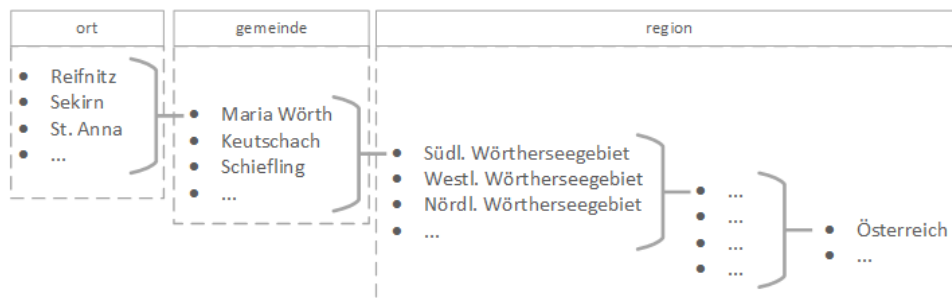


Figure 4.7: Example for spatial dboe@ema data.

Regions can be divided into topographic based regions and linguistic based regions. Both of them are formed with respect to administrative borders of the municipalities to prevent overlapping. Hence, linguistic regions depend on linguistic characteristics whereas topographical regions are defined by topographical conditions such as valleys, rivers or lakes. In the shown example, the places “Reifnitz”, “Sekirn” and “St. Anna” are located in the municipality of “Maria Wörth”. “Maria Wörth” is located in the region of “Südliches Wörtherseegebiet” (southern area of Lake Wörth). Furthermore, is this region a sub-ordinated region of the region “Wörtherseegebiet” (area of Lake Wörth). In the dboe@ema dataset, regions also handle all administrative levels above municipalities. Therefore, if looked into more detailed, some super-ordinated region of this region will be the state “Kärnten” (Carinthia), which, however, is a sub-ordinated region of the region “Österreich” (Austria).

Within GeoNames, spatial information is handled with so called feature classes and feature codes. So the usage of the correct feature class and code must be considered while searching. A list of the feature classes used by GeoNames is given in table 4.1.

The feature class “A” represents the “Administrative Boundary Features” and contains features such as nations, states and districts. When searching for municipalities, it has to be ensured that a feature of this class is searched. However, when searching for places only the feature class “P” has to be

Table 4.1: GeoNames feature classes.

Class	Description	Examples
A	Administrative Boundary Features	contry, state, region, ...
H	Hydrographic Features	lake, river, ...
L	Area Features	area, region, parks, ...
P	Populated Place Features	city, village, ...
R	Road / Railroad Features	road, railroad, ...
S	Spot Features	spot, building, farm, ...
T	Hypsographic Features	valley, mountain, region, ...
U	Undersea Features	undersea
V	Vegetation Features	forest, heath, ...

Table 4.2: GeoNames feature codes.

Code	Description	Examples
ADM1	states	Styria, Carinthia, ...
ADM2	districts	Graz, Graz-Umgebung, ...
ADM3	municipalities	Seiersberg, Pirka, ...
ADM4	fourth-order adm. division	a subdivision of a third-order adm. division
ADM5	fifth-order adm. division	a subdivision of a fourth-order adm. division

considered. Regions do not have one specific class, since they can be an administrative feature (“A”) , a hypsographic feature (“T”) or an area feature (“L”). Additionally, the class of “Administrative Boundary Features” can be divided by using feature codes as seen in table 4.2.

These codes come in handy to find the right administrative feature. When searching for municipalities, only the class “ADM3” has to be used.

Table 4.3 gives a comparison of the data structure of the spatial information from the dboe@ema and GeoNames how it is used in this prototype to link the data.

Hence, the dboe@ema hosts a lot of spatial information for all over Aus-

Table 4.3: Spatial information on dboe@ema and GeoNames.

dboe@ema	GeoNames
Ort (place)	Class P (Populated Place Features)
Gemeinde (municipality)	Class A (Administrative Boundary Features) and Code ADM3
Region (region)	Class A,T or L

tria and parts of the historical Habsburg empire, it is tried to find an automated solution for creating links. For crawling the GeoNames Application Programming Interface (API) for information, python scripts are used. The API offers the opportunity to search for features using different parameters. In the first place, it is looked for the name of the feature, which shows all features with a similar name to searched string. However, often multiple features have alike names. In a second iteration, it can be looked for the exact name, which only delivers features which names are equivalent to the sent string. If the result is still ambiguous, additional parameters such as feature class, feature codes, country or bounding boxes can be used. The request in listing 4.5 searches for the exact name “Maria Wörth”, which has to be located in the country of Austria (`countryBias=at`), is an administrative feature (`featureClass=A`) of the administrative level 3 (`featureCode=ADM3`).

Listing 4.5: GeoNames API search for the municipality of Maria Wörth.

```
http://api.geonames.org/searchJSON?
name=Maria Wörth&
isNameRequired=0&
maxRows=10&
username=testuser&
style=SHORT&
countryBias=at&
featureClass=A&
featureCode=ADM3
```

An example of using bounding boxes to narrow the result is given in the URL showed in listing 4.6

Listing 4.6: GeoNames API search for the place Reifnitz inside a bounding box.

```
http://api.geonames.org/searchJSON?name_equals=Reifnitz&
isNameRequired=0&
maxRows=10&
username=testuser&
type=SHORT&
countryBias=at&
featureClass=P&
east=14.07326&
west=14.27326&
north=46.71329&
south=46.61329
```

In order to calculate the bounding boxes, the spatial information stored in the `dboe@ema` can be used. For polygons it is easy to calculate the bounding-box and use it in the request. For point data a small box can be centered on the coordinates and step wise enlarged until a match is found. Although, it can occur that after using different parameters, the search result is still ambiguous or no match can be found. Therefore a manual rework is absolutely necessary and this approach can only be seen as semi-automated. For later mapping and to generate proper Linked Data, all found links need to be stored in the relational database. A new column from type `VARCHAR` is added to the tables of “ort”, “gemeinde” and “region”. These columns contain the URL pointing to the counterpart of the location that is found while searching the GeoNames data. An example for relational data about some places and the stored GeoNames-URI is given in figure 4.8.

	id	nameLang	geonames_url
	910	Radfeld	http://sws.aeonames.org/2768125/
	911	Radischen	http://sws.aeonames.org/3074929/
	912	Radwea	http://sws.aeonames.org/2768094/
	914	Raanitz	http://sws.aeonames.org/2768076/
	915	Raidina	http://sws.aeonames.org/2768071/
	919	Raanoen	http://sws.aeonames.org/2767979/
	920	Rankovice	http://sws.aeonames.org/3067033/
	921	Rannersdorf an der Zava	http://sws.aeonames.org/2767965/
	924	Rattersdorf	http://sws.aeonames.org/2767899/
	926	Strachotice	http://sws.aeonames.org/3065085/
	928	Raxen	http://sws.aeonames.org/2767829/
	934	Reinbera-Heidenreichstein	http://sws.aeonames.org/2767717/
	938	Reith im Alpbachtal	http://sws.aeonames.org/2767607/
	940	Retteneaa	http://sws.aeonames.org/2767529/

Figure 4.8: Example for geonames links stored in the relational `dboe@ema` database.

DBpedia

As written on the DBpedia website ⁴, it is a crowd-sourced community that provides structured information from Wikipedia as RDF-data on the Web. The task of interlinking `dboe@ema` data with DBpedia is almost similar to linking data to GeoNames. The first step is to find a class and a property which can be used as input to find a counterpart on DBpedia. At the beginning, it is considered to use parts of the data that describe the topic of the vouchers. Since this descriptions are incomplete and contain unstructured text in german and/or latin, as intended at the beginning. The fact that Wikipedia is an encyclopedia and `dboe@ema` is a corpus intended to

⁴<http://wiki.dbpedia.org/about>

facilitate dictionary compilation both use headwords. Wikipedia rather than wikitionary is chosen due to the fact that the content of dboe@ema is rather encyclopedic than just lexical. The second consideration is to directly use the lemma-information. More specifically, the “Standard German” attribute of the dialect lemmas is used. Standard German is the standardized written language in the german language region. Since Wikipedia hosts a lot of diverse information and information for just one keyword is requested, the results are ambiguous most of the time. Only the results where exactly one Wikipedia resource is returned, could be considered as correct. Thats why, like with GeoNames, manual data rework is essential. An example for some dialect lemmas and their DBpedia links are given in figure 4.9. GeoNames and DBpedia are only two of many LOD sources. As already mentioned those two are some kind of nucleus of the Linked Data cloud. If a lemma gets linked to DBpedia or a place, municipality, region to GeoNames the dboe@ema is connected to the Linked Data cloud through these two major-data-sources.

id	dbo	hochdeutsch	dbpedia_url
27904	Ouära	Ouark	http://dbpedia.org/resource/Ouark
27905	Toofen	Toofen	http://dbpedia.org/resource/Toofen
28236	Hals	Hals	http://dbpedia.org/resource/Hals
28340	Neüner	Neuner	http://dbpedia.org/resource/Neuner
28343	Gana	Gana	http://dbpedia.org/resource/Gana
28345	Spatz	Spatz	http://dbpedia.org/resource/Spatz
28351	Nußdorf	Nußdorf	http://dbpedia.org/resource/Nu%C3%9Fdorf
28482	Lack	Lack	http://dbpedia.org/resource/Lack
28542	Sinn	Sinn	http://dbpedia.org/resource/Sinn
28866	Áneis	Anis	http://dbpedia.org/resource/Anis
28897	Strâße	Straße	http://dbpedia.org/resource/Stra%C3%9Fe
28901	Palatschinke	Palatschinke	http://dbpedia.org/resource/Palatschinke
28989	Wein	Wein	http://dbpedia.org/resource/Wein
29010	Pëin	Bein	http://dbpedia.org/resource/Bein
29024	Dörfler	Dörfler	http://dbpedia.org/resource/D%C3%B6rfler
29041	Pl·t	Blut	http://dbpedia.org/resource/Blut

Figure 4.9: Example for DBpedia links stored in the relational dboe@ema database.

4.5 DATA MAPPING

For mapping the data from the dboe@ema to RDF and to provide the possibility to query it with SPARQL Protocol and RDF Query Language (SPARQL), the D2RQ-platform⁵ is used. This platform is a collection of different tools that provide SPARQL access, a Linked Data server, a RDF dump generator, a Hypertext Markup Language (HTML) interface, and a Jena API access to relational databases.

⁵<http://d2rq.org/>

4.5.1 Mapping Language

The mapping is done by using the “D2RQ Mapping Language”. With this language, specific rules about how tables and attributes are mapped to classes and properties, are created. These rules are saved in a mapping file, which is a RDF-document in turtle syntax. According to the documentation, the mapping defines a virtual RDF graph that contains information from the database. It is also said that this is similar to the concept of views in Structured Query Language (SQL), except that the virtual data structure is an RDF-graph instead of a virtual relational table. For the `dboe@ema` two mapping files are created, hence, D2RQ does not support GeoSPARQL requests and that the spatial data needs to be dumped and stored separately in a spatial triplestore. In principle, a D2RQ mapping file consists of three major parts:

1. Namespace definitions
2. Database connection
3. Creation of RDF resources and properties

Namespace definitions

This part contains all namespaces and the according prefixes that are used in the mapping. Listing 4.7 shows some prefixes that are used in this project.

Listing 4.7: Namespace definitions in a mapping file.

```
@prefix map: <#> .
@prefix db: <> .
@prefix dboe: <http://localhost/ontology/dboe#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix d2rq: <http://www.wiwiiss.fu-berlin.de/suhl/bizer/D2RQ/0.1#> .
@prefix jdbc: <http://d2rq.org/terms/jdbc/> .
@prefix geo: <http://www.opengis.net/ont/geosparql#> .
@prefix geof: <http://www.opengis.net/def/function/geosparql/> .
@prefix sf: <http://www.opengis.net/ont/sf#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
```

The namespace with the prefix map is the namespace of the mapping file and does not appear in mapped data.

Database connection

The next part of the file handles the Java Database Connectivity (JDBC) database connection and is defined by the command `d2rq:Database`. The properties of `d2rq:Database` are given in table 4.4.

In listing 4.8, a connection to a MySQL database is configured.

Listing 4.8: Example for a database definition in a mapping file.

```
map:database a d2rq:Database;
  d2rq:jdbcDriver "com.mysql.jdbc.Driver";
  d2rq:jdbcDSN "jdbc:mysql://localhost:3306/dboe_lite";
  d2rq:username "root";
  d2rq:password "";
  jdbc:autoReconnect "true";
  jdbc:zeroDateTimeBehavior "convertToNull";
  d2rq:dateColumn "belegzettel_beleg.period_start";
  d2rq:dateColumn "belegzettel_beleg.period_end";.
```

The column type of special database columns must be defined here as well. In this example, two date-columns (`d2rq:dateColumn`) are specified.

Creation of resources and properties

The last part of the file consists of various mapping definitions. If a table should be mapped, a `d2rq:ClassMap` has to be created, which maps a table to a class. Additional informations such as the data storage, the URI pattern, the class definition and the label are needed in addition. The data storage should be the already defined database connection. The URI pattern identifies instances of this class map and is a string that later is used to generate the URI of the class instances.

Table 4.4: Properties of `d2rq:Database` according to the D2RQ online documentation “The D2RQ Mapping Language” (2012).

property	description
<code>d2rq:jdbcDSN</code>	The database URL.
<code>d2rq:jdbcDriver</code>	The database driver class name.
<code>d2rq:username</code>	Database username.
<code>d2rq:password</code>	Database password.
<code>d2rq:resultSizeLimit</code>	A limit clause to all generated SQL queries.
<code>d2rq:fetchSize</code>	Number of rows to retrieve with every database request.
<code>d2rq:startupSQLScript</code>	URL of a SQL script to be executed on startup.
<code>d2rq:textColumn</code> <code>d2rq:numericColumn</code> <code>d2rq:dateColumn</code> <code>d2rq:timestampColumn</code> <code>d2rq:timeColumn</code> <code>d2rq:binaryColumn</code> <code>d2rq:booleanColumn</code> <code>d2rq:bitColumn</code> <code>d2rq:intervalColumn</code>	These properties are used to declare the column type of database columns and affects the kind of SQL literal that D2RQ will use to query for values in this column what is important for SPARQL queries and filter.

An example for defining a class map dedicated to the `dboe@ema` class of “region” can be seen in the listing 4.9:

Listing 4.9: Example for defining a class map.

```
# Tabelle region
map:region a d2rq:ClassMap;
  d2rq:dataStorage map:database;
  d2rq:uriPattern "region/@@region.id@@";
  d2rq:class dboe:region;
  d2rq:classDefinitionLabel "region";.
```

A string with the name of the class and the unique Identifiers (IDs) of the table, divided by a backslash, is used as URI pattern. Additionally the ontology class and a definition label are defined. After a class is designed, the associated property bridges can be created. In OWL, classes always consist of data properties or object properties. Data properties relate individuals to literal data such as text or numbers. In a relational database, this would be represented by normal attributes or primary keys. For these properties, the property bridge consist of a `d2rq:ClassMap` that says which class it belongs to, a definition in the ontology and a literal value provided by a column in a database-table or a SQL-query. Additional informations are the data type and the label. Listing 4.10 shows the mapping for the name

(“nameLang”) attribute in the places (“ort”) table.

Table 4.5: Properties of `d2rq:ClassMap` according to the D2RQ online documentation “The D2RQ Mapping Language” (2012).

property	description
<code>d2rq:dataStorage</code>	Reference to a <code>d2rq:Database</code> .
<code>d2rq:class</code>	An RDF Schema (RDFS) or OWL class.
<code>d2rq:uriPattern</code>	URI pattern that will be used to identify instances.
<code>d2rq:uriColumn</code>	A database column containing URIs.
<code>d2rq:uriSqlExpression</code>	A SQL expression that generates the URI.
<code>d2rq:bNodeIdColumns</code>	A comma-separated list of column names.
<code>d2rq:constantValue</code>	Single instance value.
<code>d2rq:contains Duplicates</code>	Information from not fully normalized tables.
<code>d2rq:additional Property</code>	Adds an additional property to all instances of this class.
<code>d2rq:condition</code>	SQL WHERE condition.
<code>d2rq:classDefinition Label</code>	Specifies a label for all associated class definitions.
<code>d2rq:classDefinition Comment</code>	Specifies a comment for all associated class definitions.
<code>d2rq:additionalClass DefinitionProperty</code>	Additional Property for all associated class definitions.

Listing 4.10: Example for defining data properties.

```
map:ort_nameLang a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:ort;
  d2rq:property dbo:hatNameLang;
  d2rq:propertyDefinitionLabel "ort_nameLang";
  d2rq:column "ort.nameLang";
  d2rq:datatype xsd:string;.
```

Object properties relate individuals to other individuals. In a relational database, this would be represented by “one to many” and “many to many” relationships, realized by the usage of foreign keys and association tables. In D2RQ, these foreign keys are handled with joins. The listing 4.11 shows the mapping for the n:m relation between sources (“quelle”) and their places (“ort”) of origin in the association table “quelle_lokation_liste”.

Listing 4.11: Example for defining object properties.

```
map:quelle_lokation_liste_ort_id_ref a d2rq:PropertyBridge;
  d2rq:belongsToClassMap map:ort;
  d2rq:property dbo:istLokationOrtVon;
  d2rq:refersToClassMap map:quelle;
  d2rq:join "quelle_lokation_liste.ort_id=>_ort.id";
  d2rq:join "quelle_lokation_liste.quelle_id=>_quelle.id";
```

Table 4.6: Extract of `d2rq:ClassMap`.

property	description
<code>d2rq:belongsToClassMap</code>	Specifies that the property bridge belongs to a <code>d2rq:ClassMap</code> .
<code>d2rq:property</code>	Property that connects the <code>ClassMap</code> with the object or literal created by the bridge.
<code>d2rq:refersToClassMap</code>	Properties that correspond to a foreign key on a different <code>Class Map</code>
<code>d2rq:uriColumn</code> <code>d2rq:uriPattern</code> <code>d2rq:uriSqlExpression</code>	Properties where the value is a URI.
<code>d2rq:column</code>	Properties with literal values.
<code>d2rq:sqlExpression</code>	Literal values from the results of a SQL expression.
<code>d2rq:datatype</code>	RDF datatype of literals.
<code>d2rq:lang</code>	Language tag of literals.
<code>d2rq:join</code>	Literals from other tables respectively class maps. Used when foreign keys and associative tables appear.

Some of the common used properties, taken from the D2RQ online documentation ⁶ can be seen in table 4.6.

4.5.2 Spatial data

First of all, a closer look at the part of the `dboe@ema` database that handles spatial information and geodata is taken. As already mentioned, it holds three types of spatial information: place (“Ort”), municipality (“Gemeinde”) and region (“Region”). As seen in figure 4.2, each of these three tables has an own table containing the geometry in form of point and polygon data and metadata about the geodata. This structure comes in handy when it is about to create a separated RDF dump, because one can easily dump the tables that contain the geodata and later on link it to the `dboe@ema` data. Hence, D2RQ can not handle geometries natively, they have to be mapped as a SQL-expression. The geometries then can be dumped, for example, as Well Known Text (as defined by Simple Features or ISO 19125) (WKT). Listing 4.12 shows the D2RQ syntax to create a data property from a geometry column.

⁶<http://d2rq.org/d2rq-language>

Listing 4.12: Example for mapping geometries as data property.

```
map:GISort_the_geom a d2rq:PropertyBridge;  
  d2rq:belongsToClassMap map:GISort;  
  d2rq:property geo:asWKT;  
  d2rq:propertyDefinitionLabel "GISort_the_geom";  
  d2rq:sqlExpression "ST_ASWKT(GISort.the_geom)";  
  d2rq:datatype geo:wktLiteral;
```

The SQL-Query returns the geometries, transformed to WKT. After the mapping file is created, the spatial data can be dumped. For this purpose the “dump-rdf” tool, which is part of the D2RQ platform, can be used. This tool extracts the data from the database into one RDF-file. To accomplish this, the tool needs the output format, the base URI, the output file and the path to the mapping file. The output format can be “TURTLE”, “RDF/XML”, “RDF/XML-ABBREV”, “N3” or “N-TRIPLE” which works best for large databases according to the documentation⁷. The base URI must be matched with the one later used by the triplestore.

4.5.3 Lexical data

The remaining, non-spatial `dboe@ema` tables can be provided as Linked Data using the D2RQ Server. D2RQ Server requires only the path to the mapping file as starting parameter. Once started, the data can be accessed by a web browser, a RDF-browser or the included SPARQL endpoint. The default base URI where D2R Server is running is `http://localhost:2020/`. This URI can be changed by changing the `serverBaseURI` parameter before starting the software. After the mapping task is done, the data structure, as seen in 4.10 is achieved.

4.6 CREATION A TESTING ENVIRONMENT

The prototype consists of many different tools that together form a Linked Data source, containing the `dboe@ema` lexical data. This data than can be used as basis for Linked Data applications or queries. The following list gives an overview on what type of software is used in this experiment and how to generate Linked Data from a spatial relational database.

- Relational database management system
- Ontology editor
- Mapping tool

⁷<http://d2rq.org/dump-rdf>

- Spatial triplestore
- Webservice

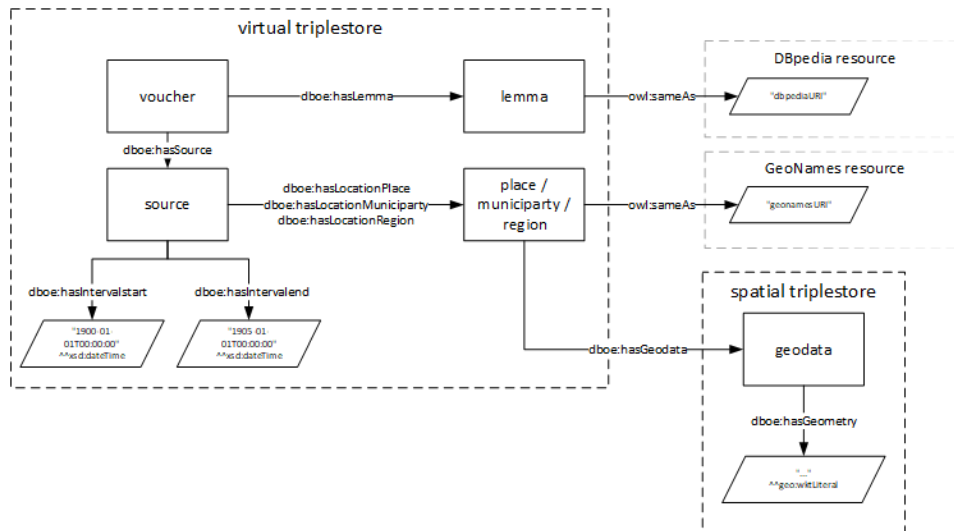


Figure 4.10: Simplified data structure of the dboe@ema Linked Data approach.

4.6.1 Relational database management system

Hence the dboe@ema uses MariaDB, a fork of the widely used MySQL Database Management System (DBMS), it is considered to use one of these two, to store the database dump at an experimental environment such as a virtual machine or a server. MariaDB is part of the XAMPP⁸ open-source cross-platform web server solution stack which contains additional useful software as we will see later on.

4.6.2 Ontology Editor

Starting from the relational database schema, an ontology gets created by using specified assignment rules. Such an OWL file can be created with any text editor. There are also ontology editors whose additional functions bring considerable simplification into the process. The user doesn't have to dispute with the syntax or the structure of the file. In addition, such tools provide more structured overview of large ontologies. This is done by

⁸<https://www.apachefriends.org/de/index.html>

dividing parts of the file in different tabs and give a hierarchic structure to properties/classes and their child elements. For this prototype, Protégé is used. Protégé is open source software and is developed at Stanford University. Protégé allows to manage the ontology Internationalized Resource Identifiers (IRIs), create classes, properties and even individuals. Additional help is the included reasoner tool. The reasoner checks if your project represents a proper OWL/RDF-File. Some additional features of “Protégé” are: simple import for external ontologies, visualization in tabular or graphical form, and different output formats.

4.6.3 Mapping Tool

D2RQ⁹ is a platform for mapping data from a relational database to Linked Data. This software is the core of this prototype. It is an all in one solution for fast providing and publishing Linked Data on the web. Its only drawback is that spatial data is not supported at this time. A workaround to solve this problem is given in this thesis. Hence federated queries are necessarily needed for this approach, it has to be made sure that this service works with the used version. During the creation of this prototype, the only version from the “d2rq.org” GitHub-branch was able to handle the SERVICE command which is used to perform federated queries. It is also worth mentioning that D2RQ is open-source software.

4.6.4 Spatial triplestore

There are many different spatial triplestores available but not all use the same standards. Criteria for selecting a product is the support of the GeoSPARQL standard and the possibility to do federated queries. It is also advantageous if a GeoSPARQL endpoint and an RDF-Explorer are included. Based on these criteria, Strabon¹⁰ is selected as spatial triplestore for this project. Strabon is an open-source spatiotemporal RDF store that is developed by the University of Athens. It is important to note that these programs often require other software to function. It is necessary to install a Java Servlet Container to run Strabon. It is also required to install a spatial DBMS to store its data. A Java Servlet Container is included in the XAMPP package and has the name “Apache Tomcat”. As DBMS PostgreSQL¹¹ with the spatial PostGIS¹² extension is used.

⁹<http://d2rq.org>

¹⁰<http://www.strabon.di.uoa.gr>

¹¹<https://www.postgresql.org>

¹²<http://postgis.net>

4.6.5 *Linked Data front end*

Triplestores store data as raw RDF-triples. In order to make this data accessible to users that want to explore the data with a HTML or RDF browser some kind of web-based software is needed. The software identifies the client and then processes the triples to serve the desired data formatting. All in one, Linked Data solutions such as D2R-Server have a Linked Data front end included to their package. Clients are directed to premade interfaces. However, these interfaces are sometimes not implemented or do not please the requested requirements. Then a third party front-end application can be used to access the data. For this prototype Pubby footnote <http://wifo5-03.informatik.uni-mannheim.de/pubby/> is used to enhance the Strabon user interface. Although Strabon offers a web application to query the data via GeoSPARQL, it lacks the possibility to simple explore the spatial data in HTML form or with a RDF-browser. Pubby uses SPARQL `DESCRIBE` queries to gather information from the SPARQL endpoint. For Pubby to work, the SPARQL endpoint URL and the URI prefix for the published resources must be defined in a configuration file. To raise the cognition-level of the served geodata, the Pubby interface can be enhanced by adding an interactive map. Pubby is a Java Servlet which creates HTML pages in real time. The creation of the website is thereby done by the Java-based template engine Apache Velocity. To manipulate the resulting website, the template files have to be manipulated. In Pubby, the datasets are showed in a table which is created by the file `proptable.vm`. Configurations to extend this table must be done in this template file. The added code checks if the data is a literal and further on contains the WKT Keywords `POINT` or `POLYGON`. If this conditions are true, the map can be created by using the web mapping JavaScript library leaflet¹³.

4.6.6 *Web server*

Web applications are hosted by the web server. The server manages the client communication and the connections to the data sources. The requirements for such a web server are HTML web page hosting and PHP functionality to enable server-sided communication to a relation database or a SPARQL endpoint. For the prototype the Apache web server included in “XAMPP” is used. The web server can also be used to provide the ontology to the clients.

¹³<http://leafletjs.com>

4.6.7 Overview

This subsection will give an overview of the used Software and how they work together in this prototype. Figure 4.11 shows the structure of the prototype. It can be seen that a client can access the Linked Data by using a Linked Data application stored on the Apache web server or browse it by using the HTML interface included in D2RQ and Strabon. The ontology is also located on the web server and can be browsed directly or as redirection form the HTML interfaces. Applications that use SPARQL/GeoSPARQL, query the D2R Server endpoint and the additional spatial data is requested in form of federated queries. A RDF browser can also be used to explore the data which is handled by the triplestore front end natively.

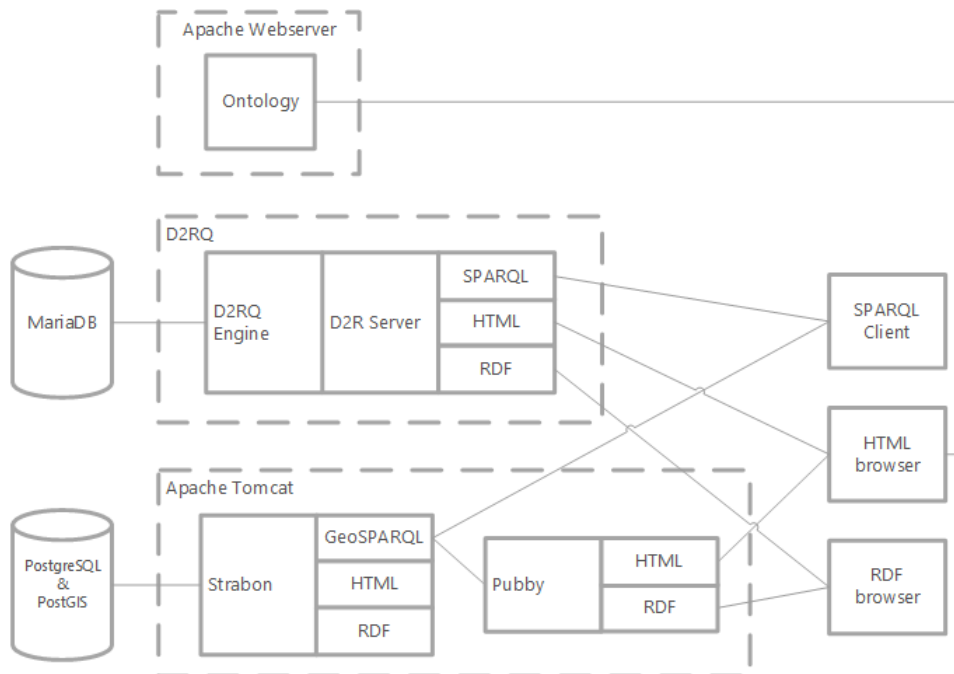


Figure 4.11: Simplified system structure of the dboe@ema Linked Data approach.

4.7 LINKED DATA APPLICATIONS

This section will give some examples on what an application, which uses the created Linked Data prototype, will look like and how it communicates with the data source.

4.7.1 Query Tool

In this example, a web application is created, which can handle federated GeoSPARQL Queries and presents the results in a table containing the RDF data and in an interactive map showing geo-features and their attributes. The SPARQL-endpoint that is used by D2RQ is capable of URL request. The SPARQL endpoint can be used through URL commands and parameters. A graphic representation of this issue is shown in figure 4.12. To get data from the SPARQL endpoint, a URL-request is needed. This

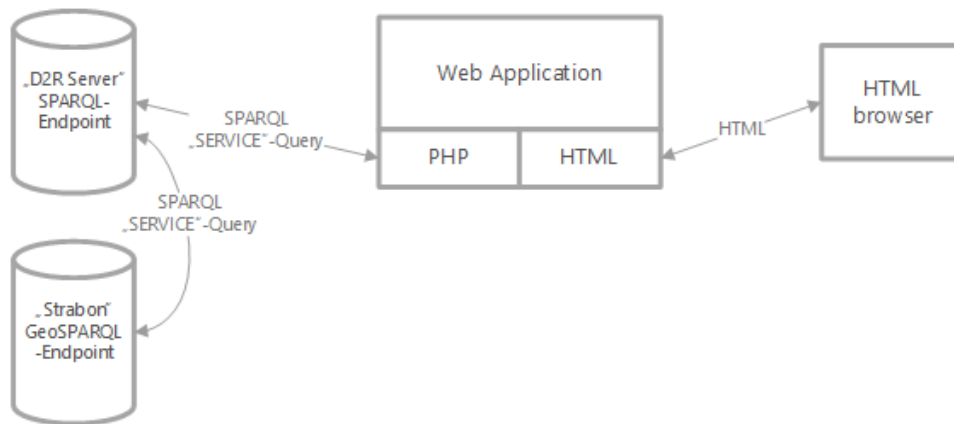


Figure 4.12: Simplified structure of the application for querying the triplestores.

can be done on the server side by using Client for URLs (CURL). In PHP: Hypertext Preprocessor (PHP) this is implemented by the “libcurl” library, which has to be installed and/or activated in the `php.ini` file. An URL-SPARQL query has the structure, which is showed in listing 4.13.

Listing 4.13: URL SPARQL Query.

```
http://localhost:2020/sparql?query= ... &format=json
```

The first part of the URL shows the address of the SPARQL endpoint. The keyword “query” marks the query that has to be executed. This is followed by the keyword “format” which is, for example, JavaScript Object Notation (JSON) since JSON is easy to handle with JavaScript based web sites. After the data is received, it is parsed and processed to be shown as table and interactive map. Since the spatial triplestore provides the geometries as WKT there is the need to transform them to a format that can be handled by the web mapping JavaScript library of choice. Leaflet¹⁴, for example, is able to draw GeoJSON features. For transforming WKT ob-

¹⁴<http://leafletjs.com>

jects to GeoJSON objects, a JavaScript library called “WICKED”¹⁵ can be used. The appearance of the map can be adapted by using the functions that are given by the web mapping JavaScript library. Different layers can be generated or an interaction with the “geo features” and pop-ups can be implemented. These functions differ from library to library and can be found in the documentations of these.

4.7.2 Search Tool

In this section, an other example for a Linked Data application is given. The tool allows the user to search the Linked Data for keywords and shows the results in form of Linked Data resources in a simple HTML page. The fact that the data is stored in a MySQL database is very helpful, hence, MySQL has a simple built-in full-text search functionality. In order to use this functionality, the tables and the columns that should be search-able must be indexed in advance. This indexing is done by the command seen in listing 4.14.

Listing 4.14: Enabling columns for full-text-search ability.

```
ALTER TABLE dboe_lite.lemma ADD FULLTEXT('hochdeutsch', 'wbo', 'dbo')
```

It shows the table “lemma” and its columns “hochdeutsch”, “wbo” and “dbo”. After the indexing process is done, it is possible to run full-text queries against text data. An example query where the table and the indexed columns are searched for the keyword “Zwiebel” is given in listing 4.15. As result the identifier of the found tuples are returned.

Listing 4.15: Searching for a keyword.

```
SELECT id FROM dboe_lite.lemma WHERE MATCH('hochdeutsch', 'wbo', 'dbo')  
AGAINST ('Zwiebel');
```

As seen in figure 4.13 the web server uses PHP to communicate with the “MySQL” database. The database is queried for the IDs of the individuals, found by the given keywords. As in section 4.5 explained, these ID are also used as part of Linked Data URI mapped by D2RQ. This information in IDs can be translated to a URI pointing to the Linked Data resource.

The so called “Inlineframe” is used to show the Linked Data in the web application. This HTML technique allows to embed another web site in the actual web site. Therefore, it is possible to show the link to a Linked Data resource and the associated HTML page given by the Linked Data front end on the same page.

¹⁵<https://github.com/arthur-e/Wicket>

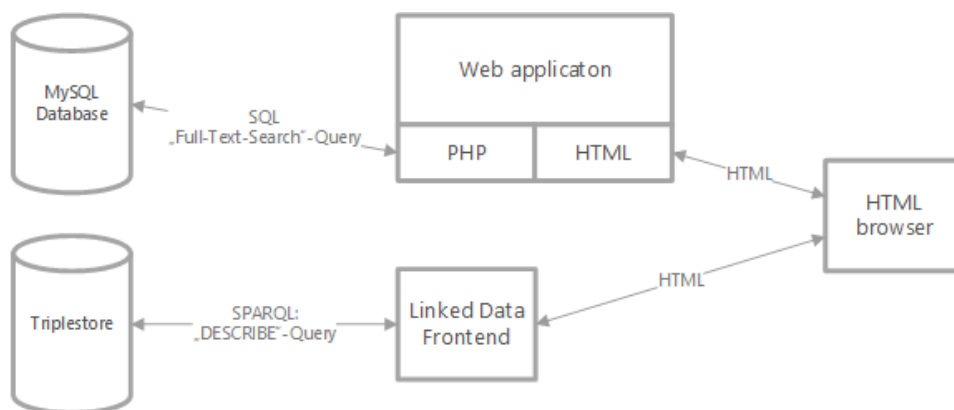


Figure 4.13: Simplified structure of the application for searching and showing linked data.

CHAPTER 5

RESULTS OF THE PROJECT

In this chapter, the results of the experiment and the answers to the scientific question that are asked in the introduction will be given.

5.1 PROTOTYPE

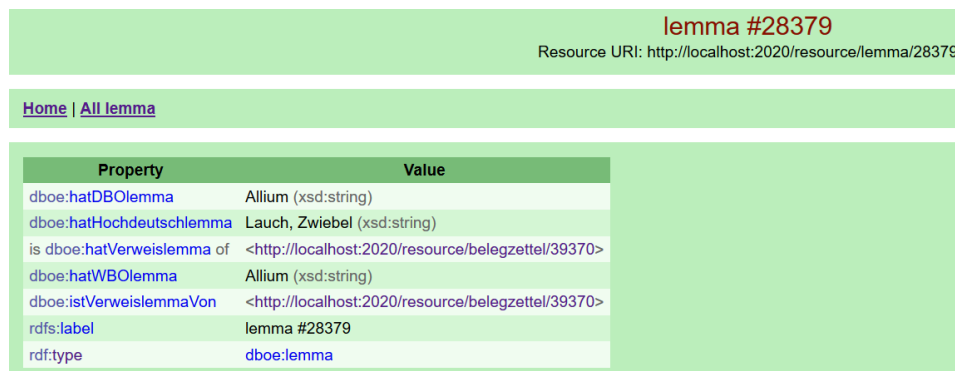
The result of the project is a working prototype that handles spatial-temporal Linked Data which origins in the “Datenbank der bairischen Mundarten in Österreich electronically mapped” (dboe@ema) database. The outcome of the single tasks, which are carried out through the whole process, are shown and analyzed. Due to demonstrate the possibilities of the prototype, two example applications are created in the course of this project, which are also presented here.

5.1.1 Ontology

The first result of this project is an ontology for the dboe@ema data. The ontology is created with Protégé by using transformation rules regarding data stored in relational databases, since the entity relationship model is an abstraction of the universe of discourse. During the whole modeling process, the in Protégé included “Hermit” reasoner was active. This tool checks if the ontology is well formed and no inferences are found in the classes or properties. After the modeling, the ontology can be saved to a file. The turtle syntax is a very common file format to export ontologies. Not all tables of the dboe@ema are considered for this project. An ontology can also be seen as graph. The graph version of the ontology is given in appendix B.

5.1.2 Mapping Server

An important part of this resulting prototype is the mapping from a relational database to a Resource Description Framework (RDF) structure. The result of the mapping- and configuration-process is a working D2R server that can handle SPARQL Protocol and RDF Query Language (SPARQL) queries and provides Linked Data in Hypertext Markup Language (HTML) or RDF form to the user or client. A screenshot showing a resource in the D2R server HTML interface is given in figure 5.1.



Property	Value
dboe:hatDBOlemma	Allium (xsd:string)
dboe:hatHochdeutschlemma	Lauch, Zwiebel (xsd:string)
is dboe:hatVerweislemma of	< http://localhost:2020/resource/belegzettel/39370 >
dboe:hatWBOlemma	Allium (xsd:string)
dboe:istVerweislemmaVon	< http://localhost:2020/resource/belegzettel/39370 >
rdfs:label	lemma #28379
rdf:type	dboe:lemma

Figure 5.1: Screenshot of a resource, represented by the D2R-Server HTML interface.

5.1.3 Spatial triplestore

Hence the fact, that D2RQ cant handle spatial queries, a spatial triplestore is needed. In order to fill the spatial triplestore, the spatial data first is dumped from the MySQL database to an RDF file by using the D2RQ mapping tool. The data is stored in the “Strabon” triplestore, which provides a GeoSPARQL endpoint for performing queries against the spatial triples. In figure 5.2 an example query is given by using the Strabon HTML interface.

Additionally, a front end is implemented to enable data exploring with a custom interface, containing a interactive map. This is achieved by using the Java servlet Pubby and applying some modifications to its HTML templates. The result of this work can be seen in figure 5.3.

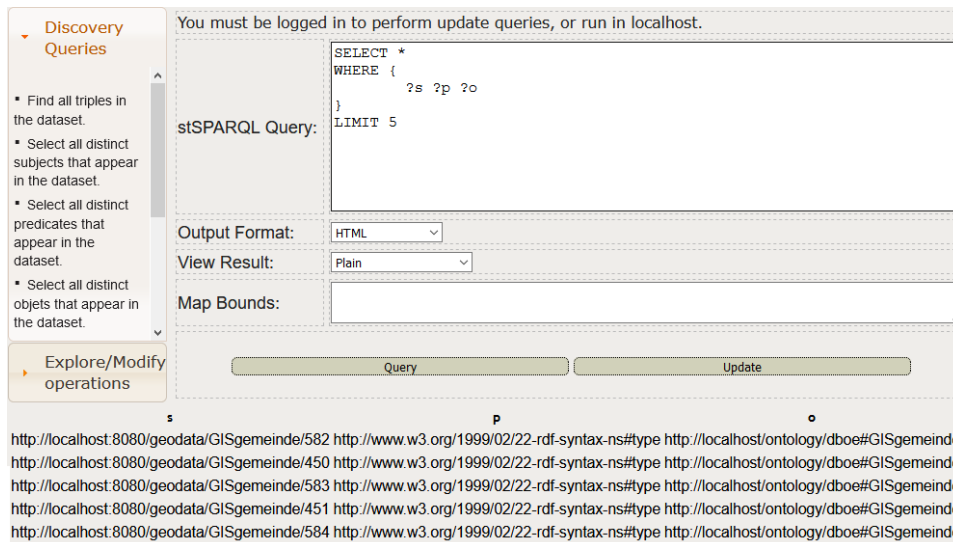


Figure 5.2: Screenshot of an example query showing the first five triples, stored in the Strabon triplestore.

5.1.4 Query Tool

The intention of creating this demo application is to show the possibilities that SPARQL queries bring to analyzing the data. The layout of the application can be seen in figure 5.4. It consists of a text area, in which queries can be entered. An additional text area and a drop-down menu for naming and coloring possible geometries are part of the interface that can be found. The results of a query are then shown in a table. If the results contain Open Geospatial Consortium (OGC) simple features in form of POINT, POLYGON or MULTIPOLYGON, a map with appropriate layers is created. Each query can be added as a new layer with a custom name and color to the map. This allows the user to compare the spatial change over time and/or other parameters in one map.

In this particular case, as seen in figure 5.4, the results for a spatiotemporal query that gives all municipalities where a dialect word for carrot was recorded, but it was different from the word “Karottn”. The results are settled in the year 1916 and shown as green polygons on the map. Figure 5.5, on the other hand, shows how the dialect word “Karottn” is spreading and replacing other dialect words as it shows the results for the same query but in the year 1966 represented by the red layer.

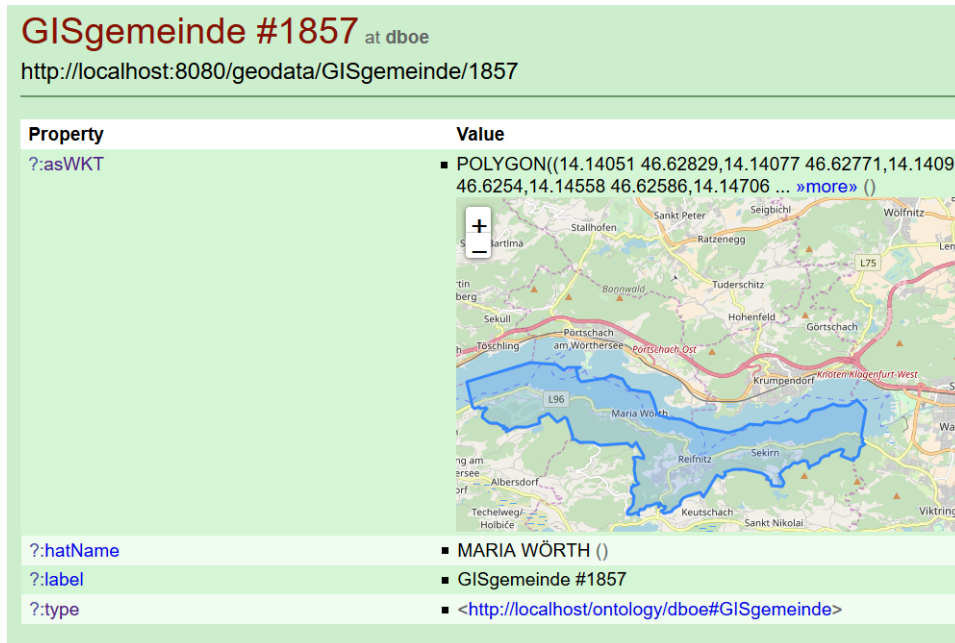


Figure 5.3: Screenshot of a geospatial resource, represented by the enhanced Pubby interface.

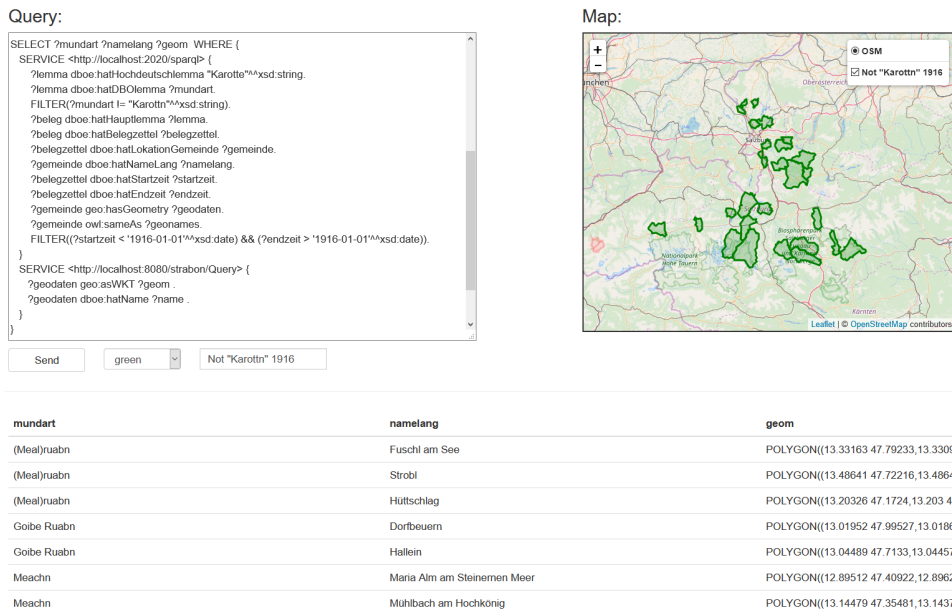


Figure 5.4: Demonstration of the query tool (part 1).

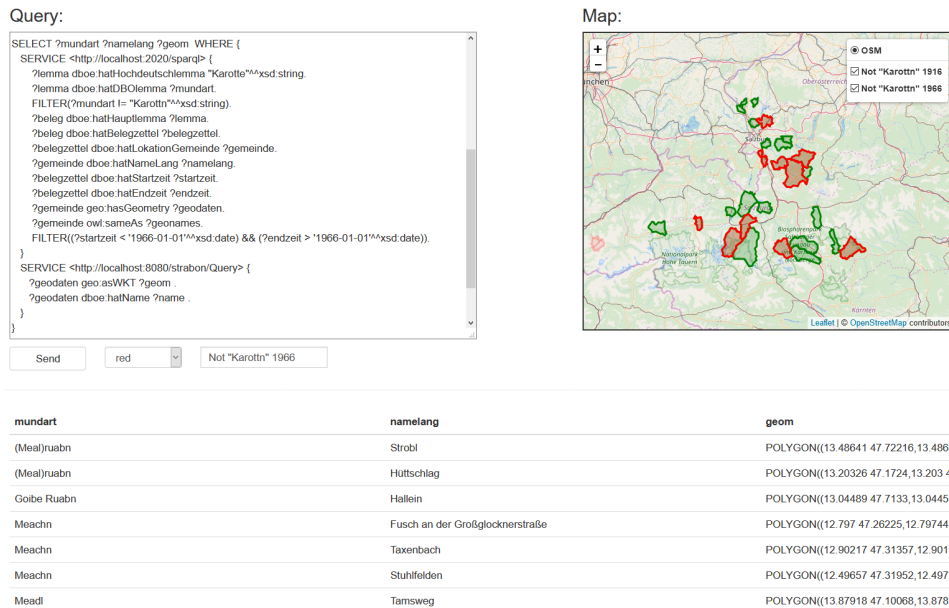


Figure 5.5: Demonstration of the query tool (part 2).

Please note that the temporal data for this example is only fictional and is inspired by the “Sprachatlas Salzburg”¹, a website that allows users to listen to dialect words of some Salzburg municipalities and how they evolved by comparing by two generations of people. Although the data does not directly origin in the dboe@ema, it gives a good insight into the possibilities of the prototype and the power of federated spatial-temporal queries.

5.1.5 Search Tool

The implemented search tool for the dboe@ema data should make it easier for users to find specific data and explore it. To do so, the user can search for keywords and follow the links that are provided by the resulting properties, to discover new information or Linked Data. The application consists of an input mask on the top of the page where the user can enter one or more keywords. These words are searched for in the selected class/table. The results are represented as links to the resource in which corresponding text is found. The bottom of the page holds a frame that shows the HTML page of the resource, on which the user clicked on.

Figure 5.6 shows the results for the search of a place called “Reifnitz”. The table contains two resources in which properties the keyword is found.

¹<https://www.sprachatlas.at/salzburg/>

By clicking on the resource the HTML interface of the D2R-Server on which the resource is stored is shown.

The screenshot shows a search interface with a search box containing 'Reifnitz', a dropdown menu set to 'place', and a 'Send' button. Below the search bar, the results section displays two URIs: <http://localhost:2020/resource/ort/2919> and <http://localhost:2020/resource/ort/4628>. A green bar indicates the selected 'Resource URI: http://localhost:2020/resource/ort/2919'. Below this, there are navigation links for 'Home' and 'All ort'. The main content is a table with two columns: 'Property' and 'Value'.

Property	Value
geo:hasGeometry	<http://localhost:8080/geodata/GISort/1605>
dboe:hatBearbeitungsgebiet	<http://localhost:2020/resource/bearbeitungsgebiet/2>
dboe:hatNameKurz	Reifnitz (xsd:string)
dboe:hatNameLang	Reifnitz (xsd:string)
is dboe:hatOrt of	<http://localhost:2020/resource/gemeinde/1854>
is dboe:istBearbeitungsgebietVon of	<http://localhost:2020/resource/bearbeitungsgebiet/2>
dboe:istOrtIn	<http://localhost:2020/resource/gemeinde/1854>
rdfs:label	ort #2919
owl:sameAs	<http://sws.geonames.org/2767732/>
rdf:type	dboe:ort

Figure 5.6: Demonstration of the search tool.

By clicking further links, the user can explore additional information such as the geodata that is related to the place and stored in the Strabon triplestore. It gets visualized by the customized Pubby HTML interface. The output of accessing the spatial data through the search tool is given in figure 5.7.

The dataset from 5.6 is also related to a GeoNames resource, which can be called by just pressing the appropriate link. As result, the GeoNames user-interface for the requested individual shows up in the bottom part of the page and can be seen in figure 5.8.

It should be taken into account that the search tool searches the MySQL tables of the relational dboe@ema database and not the Linked Data triples. The links to the RDF resources are created from the IDs that are given from the tuples that are found by the MySQL Full-text-search function. The search only works for columns of the type varchar that are indexed with the appropriate SQL command.

Reifnitz place Send

Results

<http://localhost:2020/resource/ort/2919>

<http://localhost:2020/resource/ort/4628>

Property	Value
?:asWKT	POINT(14.179185451323 46.605318274566)
?:hatName	Reifnitz
?:label	GISort #1605
?:type	< http://localhost/ontology/dboe#GISort >

Figure 5.7: Geodata accessed through the search tool.

Reifnitz place Send

Results

<http://localhost:2020/resource/ort/2919>

<http://localhost:2020/resource/ort/4628>

GeoNames About Browse Download API Help Paris, Mount Everest, Nev Q x +

Reifnitz ca. 442 m
 P PPL populated place **2767732**
 Austria AT » Carinthia 02 » Klagenfurt Land 204 » Maria Wörth 20419
 46.60722, 14.18167 N 46°36'26" E 14°10'54"

geotree .kml .rdf

Figure 5.8: Geonames resource accessed through the search tool.

5.2 INTERPRETATION

5.2.1 *Capability of the Linked Data prototype*

The triplestore system, used in this prototype and the relational database, can handle queries pretty well. SPARQL provides the same functionality as common SQL solutions. The functionality is nearly identical since SPARQL is constantly evolving and getting new features. With GeoSPARQL even OGC-conform spatial queries can be accomplished. SPARQL also has the ability to create federated queries between different data sources and request metadata information which is a huge advantage compared to the distinct model of relational Database Management System (DBMS).

When it comes to performance, the relational database outruns the triplestores, which are used in this prototype, with ease. This has a simple, but logic reason. Non-native RDF stores like D2R Server or Strabon use relational databases to store their data. These triplestores can never bring a better performance than the DBMS on which they depend. An additional slowdown is caused by complex mapping and storing/access routines that have to be processed every time when data is requested. The bigger a dataset is the larger the gap in performance will become.

Searching for strings in the RDF data can be done with SPARQL and regular expression. This approach is quite slow since all the data has to be searched for a requested string. A similar but more efficient way to establish the search for keywords is provided by MySQL Full-Text-Search, which indexes the data for faster results. This approach is used by the presented search tool.

The question on how the triplestore solution compares to the relational database solution is strongly depending on the application and the requirements on the data. For data with a simple structure, a predictable, heavy load and the requirement for a closed system, a classical relational DBMS is the best solution. Here, Linked Data can still be published by creating an API. However, for data with a meaningful ontology, which should be linked to other data in the Semantic Web and the desire to query all the data for analysis purpose, a Linked Data approach is certainly advantageous. As projects like DBpedia or BabelNet² show, dictionaries and lexical data belong to this group. This experiment also shows that there are tools like the D2R server, which provide the possibility to provide Linked Data from relational databases which gives the possibility to use the strength of both implementations in one, custom solution.

²<http://babelnet.org>

5.2.2 *Influence on digital humanities*

This section is a theoretical section and tries to answer the question on how the developed prototype will have an influence on digital humanities by taking the attributes of it in account and validate them with a paper by Chiarcos et al. (2011) about the linguistic linked open data cloud.

Chiarcos et al. (2011) pointed out that applying Linked Data principles to lexical and other linguistic resources has a number of advantages that are attributed to the Linked Data principles described by Bizer et al. (2009) and also mentioned in this thesis at section 2.3. Chiarcos et al. (2011) give five specific advantages of modeling linguistic resources as Linked Data:

structural interoperability: This point mentions RDF as a structurally interoperable format. In a broader sense, this means that data can be combined and processed from different sources without modification to its format. Also, structural interoperability by content negotiation is described, which allows the data source to represent the data as HTML file to a normal web browser and as original RDF-data to a RDF-client.

federation: It is pointed out that federation allows queries over Linked Data that is stored in multiple different repositories physically located at different servers. In other words, resources can be merged even if these resources are physically distributed over different repositories.

conceptual interoperability: It is said that the usage of distributed can also be exploited to use shared or reference vocabularies.

ecosystems: This point refers to the existence of multiple standards and recommendations like RDF, OWL, (Geo)SPARQL, etc. that result in a large number of tools and techniques to process Linked Data. There are also active to communities like the World Wide Web Consortium (<http://www.w3.org/>) (W3C) that drive progress.

dynamic import: It is claimed that within the Linked Data approach information can be represented by a resolvable Uniform Resource Identifier (URI) which is always accessible in its latest form. This means that version control is not needed. However, this actuality of data also includes a downside. It can occur that references from external resources are no longer valid or can not be found.

Hence this prototype fulfills these five principles, it can be said that it is a positive development in the field of linguistics and especially for the dboe@ema project.

CHAPTER 6

SUMMARY AND OUTLOOK

In this chapter, a summary of this thesis and an outlook for possible extensions or tasks according to this thesis and the resulting prototype is given.

6.1 SUMMARY

The goal of this thesis was to create a spatial-temporal Linked Data prototype from a lexical data set stored in a relational database. In doing so, it is important to use the existing resources in the form of a MySQL database in the best possible way. For this purpose, an approach is shown, in which the data mostly remains in the relational Database Management System (DBMS) and is mapped by a special software to be published and queried as Linked Data. At the beginning of the project, an ontology based on the existing Entity Relationship Model (ERM) is created. The foundation for this transformation is provided by the research done by Noy and McGuinness (2001) and Myroshnichenko and Murphy (2009). In addition, the data needs to be adapted to be used in a Linked Data system. This includes data refining as well as the search and storage of links to resources from the Linked Open Data (LOD) cloud such as DBpedia or GeoNames. In order to be able to publish data from “Datenbank der bairischen Mundarten in Österreich electronically mapped” (dboe@ema) as Linked Data a D2R Server is used. This provides the possibility of publishing data from a relational database as Linked Data and query it via SPARQL Protocol and RDF Query Language (SPARQL). Since D2R Server cannot handle spatial data and its corresponding query language GeoSPARQL, the geometries are stored in a separate, spatial triplestore. In the course of the thesis, a functional prototype with two demo applications are created. The applications show how the spatiotemporal Linked Data can be queried, searched and how the received information can be displayed. The comparison of the prototype with the classical relational DBMS shows that the Linked Data approach is better in terms of decentralization and analysis capabilities. This is enabled by the Linked Data principles which provide high interoperability and federation. In terms of performance, a DBMS system is superior compared to a non native triplestores such as the ones used in this prototype, hence

these triplestores also use relational DBMS as their basic framework for data storage. What system is more suitable depends heavily on the application case. Digital humanities and especially lexical data sets benefit greatly from the Linked Data cloud and, thus, it is desirable that the dboe@ema becomes a part of it. The possibility of spatial and temporal analysis is an additional value.

6.2 OUTLOOK

Since the approach shown here is only a prototype, improvements are possible and also necessary to advance its functionality. For example, a real implementation of the prototype would be desirable. It should also be considered to integrate the dboe@ema data into the Linguistic Linked Open Data cloud. This is the only option that the full conductivity of a Linked Data approach can be guaranteed. It is also important to note that the approach used to create an ontology for this prototype is a very general one. Here, a specialization could be carried out and the ontology could be adapted to other projects of the open linguistic cloud like DBpedia. By this point, the Linked Data software lacks a fast way to map and publish spatial data stored in a relational DBMS. When it is possible to map spatial data in a way similar to D2RQ as Linked Data in the future, it would be easier for many disciplines to provide their spatial data as a LOD and to network them with other resources. A result would be the offer of more possibilities for analysis due to more open data. In the case of dboe@ema, it should also be suggested to refine the data regarding its temporal information and the attributes that should be used to create links to resources from the LOD cloud. Concerning the dboe@ema, it can also be considered that, in addition to the approach implemented here, there is another simple way to publish data as Linked Data. It is presented in subsection 2.3.6. Thereby, the data is also stored in a relational database but is called via an appropriate Application Programming Interface (API) and represented in a Linked Data format. The disadvantage here, however, is that it is a proprietary solution and that cross-SPARQL queries would not be possible. Eventually, it can be said that Linked Data is still in its inception but is constantly evolving. In this process, more and more data is freely accessible through the LOD cloud. This will create, sooner or later, the possibility to conduct comprehensive, interdisciplinary research and analysis. Spatiotemporal Linked Data will be a fixed component of this future and constitutes a great opportunity for researchers from the field of Geographic Information Science (GIScience) to establish themselves in this emerging field (Kuhn et al., 2014).

BIBLIOGRAPHY

- Araujo, S., Houben, G.-J., Schwabe, D., & Hidders, J. (2010). Fusion—visually exploring and eliciting relationships in linked data. *The Semantic Web—ISWC 2010*, 1–15.
- Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., & Ives, Z. (2007). Dbpedia: a nucleus for a web of open data. *The semantic web*, 722–735.
- Bartelme, N., Fliedl, G., Hassler, M., Mayer, H. C., Nickel, J., Scholz, J., ... Wandl-Vogt, E. (2016). Mapping Languages: Erfahrungen aus dem Projekt dbo@ema. Retrieved from http://www.hassler.world/assets/publications/2008%7B%5C_%7DDBOE%7B%5C_%7DAGIT%7B%5C_%7DHassler.pdf
- Battle, R. & Kolas, D. (2012). GeoSPARQL: Enabling a Geospatial Semantic Web. *Semantic Web Journal*, 1–17. doi:10.3233/SW-2012-0065
- Berners-Lee, T. [T.], Fielding, R., & Masinter, L. (2005). Uniform resource identifier (uri): generic syntax. Retrieved October 23, 2017, from <https://tools.ietf.org/html/rfc3986>
- Berners-Lee, T. [Tim]. (2006). Linked data - design issues. Retrieved May 30, 2017, from <https://www.w3.org/DesignIssues/LinkedData.html>
- Berners-Lee, T. [Tim], Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific american*, 284(5), 28–37.
- Berrueta, D., Phipps, J., Miles, A., Baker, T., & Swick, R. (2008). Best practice recipes for publishing rdf vocabularies. Retrieved June 20, 2017, from <https://www.w3.org/TR/swbp-vocab-pub/>
- Bitzer, J. & Schröder, P. (2005). The impact of entry and competition by open source software on innovation activity. *Industrial Organization*, 51201.
- Bizer, C., Heath, T., & Berners-Lee, T. (2009). Linked data—the story so far. *International journal on Semantic Web and Information Systems*, 5(3), 1–22. doi:10.4018/jswis.2009081901. arXiv: 1011.1669
- Brickley, D., Guha, R. V., & McBride, B. (2014). Rdf schema 1.1. Retrieved September 22, 2017, from <https://www.w3.org/TR/rdf-schema/>
- Burdick, A., Drucker, J., Lunenfeld, P., Presner, T., & Schnapp, J. (2012). *Digital humanities*. MIT Press.
- Burgos, J. L. M. (2011). Semantic Web Standards. *SNET Computer Engineering*, 1–8. Retrieved from http://www.pdfFiller.com/948565-semantic-web-standards%7B%5C_%7Dburgos-Semantic-Web-Standards---SNET-Variou-Fillable-Forms-snet-tu-berlin

- Carroll, J., Bizer, C., Hayes, P., & Stickler, P. (2005). Named graphs. *Web Semantics: Science, Services and Agents on the World Wide Web*, 3(4). Retrieved from <http://www.websemanticsjournal.org/index.php/ps/article/view/76>
- Chambers, J. K. & Trudgill, P. (1998). *Dialectology*. Cambridge University Press.
- Chen, P. P.-S. (1976). The Entity-Relationship Model - Toward a Unified View of Data. *ACM Transactions on Database Systems*, 1(1), 9–36. doi:10.1145/320434.320440
- Chiarcos, C., Hellmann, S., & Nordhoff, S. (2011). Towards a linguistic linked open data cloud: The open linguistics working group. *TAL Traitement Automatique des Langues*, 52(3), 245–275. doi:10.1007/978-3-642-31782-8
- Cyganiak, R., Wood, D., Lanthaler, M., Klyne, G., Carroll, J. J., & McBride, B. (2014). Rdf 1.1 concepts and abstract syntax. Retrieved September 22, 2017, from <https://www.w3.org/TR/rdf11-concepts/>
- Dear, M., Ketchum, J., Luria, S., Richardson, D., et al. (2011). *Geohumanities: art, history, text at the edge of place*. Routledge.
- Elmasri, R. & Navathe, S. B. (1994). *Fundamentals of database systems (2nd ed.)* Redwood City, CA, USA: Benjamin-Cummings Publishing Co., Inc.
- Group, W. O. W. (2012). Owl 2 web ontology language. Retrieved June 15, 2017, from <http://www.w3.org/TR/owl2-overview/>
- Gruber, T. R. (1993). A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2), 199–220. doi:http://dx.doi.org/10.1006/knac.1993.1008
- Hall, W. & Tiropanis, T. (2012). Web evolution and Web Science. *Computer Networks*, 56(18), 3859–3865. doi:10.1016/j.comnet.2012.10.004
- Hartig, O. (2009). Provenance information in the web of data. *LDOW*, 538.
- Heath, T. & Bizer, C. (2011). *Linked data: Evolving the Web into a global data space (1st edition)*. doi:10.2200/S00334ED1V01Y201102WBE001. arXiv: arXiv:1011.1669v3
- Hitzler, P., Krötzsch, M., Rudolph, S., & Sure, Y. (2008). *Semantic web*. Springer. Retrieved from <http://sciyo.com/books/show/title/semantic-web>
- Hoch, S. & Hayes, J. J. (2010). Geolinguistics: the incorporation of geographic information systems and science. *The Geographical Bulletin*, 51(1), 23.
- Horrocks, I., Patel-Schneider, P., & van Harmelen, F. (2003). From SHIQ and RDF to OWL: The Making of a Web Ontology Language - Google Search. *Journal of Web Semantics*, 1(1), 7–26. doi:10.1016/j.websem.2003.07.001
- Jannidis, F., Kohle, H., & Rehbein, M. (2017). *Digital humanities: eine einföhrung*. Springer-Verlag.

- Jessop, M. (2008). The inhibition of geographical information in digital humanities scholarship. *Literary and Linguistic Computing*, 23(1), 39–50.
- Kuhn, W., Kauppinen, T., & Janowicz, K. (2014). Linked Data - A paradigm shift for Geographic Information Science. *Proceedings of The Eighth International Conference on Geographic Information Science (GIScience2014)*, 173–186. doi:10.1007/978-3-319-11593-1_12
- McGuinness, D. L. & Van Harmelen, F. (2004). Owl web ontology language - overview. Retrieved May 30, 2017, from <https://www.w3.org/TR/owl-features/>
- Musen, M. A. (1992). Dimensions of Knowledge Sharing and Reuse. *Computers and biomedical research*, 25(5), 435–467.
- Myroshnichenko, I. & Murphy, M. C. (2009). Mapping ER schemas to OWL ontologies. *ICSC 2009 - 2009 IEEE International Conference on Semantic Computing*, 324–329. doi:10.1109/ICSC.2009.61
- Noy, N. F. & McGuinness, D. L. (2001). Ontology Development 101: A Guide to Creating Your First Ontology. *Stanford Knowledge Systems Laboratory*, 25. doi:10.1016/j.artmed.2004.01.014. arXiv: 1304.1186
- The D2RQ Mapping Language. (2012). Retrieved September 12, 2017, from <http://d2rq.org/d2rq-language>
- Perry, M. & Herring, J. (2012). OGC GeoSPARQL-A geographic query language for RDF data. *OGC Candidate Implementation Standard*, 57. Retrieved from <http://www.opengis.net/doc/IS/geosparql/1.0>
- Prud'hommeaux, E., Buil-Aranda, C., Seaborne, A., Polleres, A., Feigenbaum, L., & Williams, G. T. (2013). Sparql 1.1 federated query. Retrieved May 30, 2017, from <https://www.w3.org/TR/sparql11-federated-query/>
- Scholz, J., Lampoltshammer, T. J., Bartelme, N., & Wandl-Vogt, E. (2016). Spatial-temporal modeling of linguistic regions and processes with combined indeterminate and crisp boundaries. *Lecture Notes in Geoinformation and Cartography*, 133–151. doi:10.1007/978-3-319-19602-2_9
- Schreiber, G., Raimond, Y., Manola, F., Miller, E., & Brian, M. (2014). Rdf 1.1 primer. Retrieved May 30, 2017, from <https://www.w3.org/TR/rdf11-primer/>
- Sibler, P., Weibel, R., Glaser, E., & Bart, G. (2012). Cartographic visualization in support of dialectology. *Proceeding AutoCarto 2012*.

Appendix

A: Entity-relationship-model of the simplified DBOE data.	82
B: Ontology of the simplified DBOE data, generated with “NawigOwl”.	83

B: Ontology of the simplified DBOE data, generated with “NawigOwl”:

