FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Federation Solutions for Linked Data Applications

Tiago Gonçalves Gomes

Mestrado em Engenharia Informática e Computação

Supervisor: Sérgio Nunes

October 10, 2023

Federation Solutions for Linked Data Applications

Tiago Gonçalves Gomes

Mestrado em Engenharia Informática e Computação

Approved in oral examination by the committee:

Chair: Prof. Nuno Macedo
External Examiner: Prof. Nuno Escudeiro
Supervisor: Prof. Sérgio Nunes

October 10, 2023

Abstract

The semantic web is an evolving extension of the World Wide Web in which data is structured and linked in a way that enables machines to understand the meaning and context of information. Its emergence brought the concept of Linked Data, a set of principles that are revolutionizing how data is shared and reused on the World Wide Web. These principles state a set of best practices for publishing and connecting structured information on the web, motivating the appearance of various new applications. To fully take advantage of the vast amount of data on the semantic web, these applications should have data integration techniques to make their knowledge bases more understandable to their users. However, linking data from different sources and reconciling differences in the data models and vocabularies is not straightforward, often requiring manual work. Despite a large number of datasets published on the web of data, the access to information tends to be focused on a specific dataset, neglecting the full power of data integration and interoperability. Moreover, a new approach to user interaction is imperative to facilitate the integration of linked data into applications. This research aims to design and implement a solution that takes advantage of existing open knowledge graphs on the web of data to improve the meaning and expand the information of a linked data application. The solution enables entity linking with external sources and then querying and navigating across multiple knowledge bases using linked data integration techniques, namely SPARQL query federation. A linked data application in the context of classical music, Linked Classical, was developed to serve as a case study for the integration of the developed solution. The application went through a process of functional evaluation, to ensure its requirements were fulfilled. Furthermore, the solution was published as an open-source project to serve as a template for developing similar projects. Through the exploration and development of the Linked Classical application, this work seeks to contribute to the advancement of linked data technologies and promote their wider adoption across various domains. By providing a tangible example and sharing valuable insights gained throughout the development process, this research aims to empower researchers and developers to leverage the semantic web and linked data technologies, thereby unlocking new possibilities in data integration, knowledge representation, and application development.

Keywords: Semantic Web, Linked open data, SPARQL query federation

ACM Classification: CCS → Information systems → World Wide Web → Web data description languages → Semantic web description languages

Acknowledgements

I want to express my appreciation to my supervisor, Sérgio Nunes, for his invaluable support and guidance. His mentorship played a crucial role in shaping this work, and I genuinely appreciate his insights and encouragement.

I also want to extend my gratitude to my family and friends. Your belief in my abilities has been a constant driving force. Your support throughout this journey has meant the world to me.

With sincere thanks,

Tiago Gomes

Contents

List of Figures

List of Tables

Abbreviations

WWW	World Wide Web
LD	Linked Data
LOD	Linked Open Data
URI	Uniform Resource Identifier
HTTP	Hypertext Transfer Protocol
RDF	Resource Description Framework
OWL	Web Ontology Language

Chapter 1

Introduction

1.1 Context

One of the most important events of the past decades was the emergence of the Internet, the global network of interconnected computer networks that link devices worldwide. This global network, well-known by all of us, has reshaped many aspects of society, from communication, entertainment, education, and even health. While thirty years ago people had to send letters and wait days or even weeks to communicate with others that are far away, nowadays it is possible to interact with people from the other side of the world in real-time. We can watch movies, listen to music, attend concerts, read books and newspapers, be taught by the best instructors, and be seen by the best doctors in the world, all of this without leaving the comfort of our homes.

One of the pillars of the materialization of this new digital world is the World Wide Web (WWW), a network of interconnected documents that, combined with text search, can be accessed using a web browser and allow users to navigate them by following its hyperlinks. In addition to browsing the web, users can also contribute to it by adding new documents and respective links, marking a fundamental step to information access [1].

The current form of the WWW is the result of a continuous process of evolution [2]. Web 1.0 is a term used to refer to its first stage, when websites were static, only allowing the user to read the documents and not interact with them. Web 2.0 was the second stage, when user interaction became common, bringing new applications to reality, such as social networks and commerce websites. Finally, Web 3.0, also known as the Semantic Web, aims to extend the current web to give meaning and structure to the information and make it possible for both humans and machines to interact with it [3].

The concept of linked data is at the base of the semantic web, with the definition of a set of rules stating the best practices for publishing and connecting structured information on the web. Linked data led to the emergence of a global network of data known as the Web of Data [4]. The rise of the Web of Data also brought a new concept, Linked Open Data, a term used when the linked data is available for anyone to use. Like Linked Data, Linked Open Data is based on a set of principles used for improving access and management of the information [5].

With the large amounts of datasets available on the web of data, it is fundamental to ensure the interoperability between them. The ability to intermix these different datasets enables different systems and organizations to share and exchange information with each other in an easy way, giving the ability to combine them to create larger systems with even more value [6]. However, the integration between the published datasets is not easy to achieve, and many research opportunities exist under this topic [7].

1.2 Motivation

The appearance of linked data applications accompanied the rise of the semantic web. Given the increasing quantity of linked data available on the web, it is evident that these platforms could greatly benefit from using this information. The integration between the knowledge graphs supporting linked data applications and the global semantic networks would facilitate access to new information. Furthermore, it would increase the Linked Open Data Cloud, enriching the web of data [4].

A variety of different techniques can be used for the integration of data between knowledge graphs. However, the SPARQL Federation technique stands out, because of its simplicity when compared to the other approaches [7]. Nevertheless, SPARQL endpoint federation systems still have flaws and need a lot of improvement to reach their full potential. It is still not straightforward to implement a system that takes advantage of federated data. Because linked data, and in particular linked data integration, is a relatively recent topic, there are still many research opportunities in this subject [8].

Furthermore, there are still many challenges regarding the user interaction with linked data applications, especially in SPARQL federation systems. As the amount of data is generally very high, problems such as scalability, flexible user interaction, reusability, and data management complexity are a reality in the linked data visualization field [9].

This work is focused on understanding the capabilities and limitations of SPARQL endpoint federation, as well as how to overcome the implementation and user interaction challenges of systems that take advantage of this technique.

1.3 Objectives

This research aims to design and implement a solution that takes advantage of existing open knowledge graphs on the web of data to improve the meaning and expand the information present in information systems. The solution will enable entity linking with external sources and then querying and navigating across multiple knowledge bases using SPARQL query federation. The objectives of the research are as follows:

- Review the state of the art in the areas of linked data, linked data integration, SPARQL endpoint federation systems, and linked data interaction: literature review and investigation of similar projects on linked data integration and federation;

- Development of a prototype for a linked data application with federation support: design and documentation of a system that gives support to federation for linked data applications, using state of the art technology and with emphasis on the user interaction;
- Implementation and integration of the prototype in the Linked Classical platform: use the Linked Classical platform as a case-study for the developed system;
- System validation: functional evaluation of the developed prototype.

1.4 Document Structure

Besides this introduction, this document has six more chapters. Chapter 2 presents the background information and state of the art of linked data and linked open data. Chapter 3 exposes integration techniques in linked data, with focus on SPARQL Federation. Chapter 4 describes the problem and proposes a solution. Chapter 5 exposes the methodology used, with the presentation of the Linked Classical platform. Chapter 6 presents the evaluation of the Linked Classical application. Finally, a conclusion is presented in Chapter 7.

Chapter 2

Linked Data

The emergence of the semantic web brought the concept of Linked Data, a set of principles that are revolutionizing how data is shared and reused on the World Wide Web. These principles state the best practices for publishing and connecting structured information on the web. In this chapter, we provide an overview of linked data, presenting the motivation for its development (Section 2.1), principles (Section 2.2 and 2.3) and its associated standards and technologies (Section 2.4). Section 2.5 describes some linked data visualization techniques. Section 2.6 describes what are the main challenges related to the usage of linked data. Finally, Section 2.7 presents real-world applications using linked data.

2.1 Motivation

We live in a world where data is becoming more and more important, being key for many institutions of all sizes and sectors. However, the traditional approaches to data management are becoming unsuitable for the reality of the web. Firstly, the conventional approaches use proprietary databases and interfaces, making data access more difficult or impossible for the majority of the users. Furthermore, because traditional databases do not assign globally unique identifiers to their data items, it is impossible to link data across multiple databases, making it difficult to build applications that take advantage of all the data available on the web [10].

To overcome these problems, Tim Berners-Lee proposed a set of principles for publishing and connecting structured information on the web: the linked data principles [11]. Linked data aims to provide a framework and define standards for data storage, querying, and integration. The existence of standards for a global decentralized database of interconnected resources enables the appearance of new and more powerful applications, that can use all the available structured information on the web [10].

2.2 Principles

In 2006, Berners-Lee proposed a set of rules for publishing and connecting data on the web [11]:

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

These four rules became known as the linked data principles, and aim to extend the success of the web of documents. In the traditional web, documents also have URIs, namely HTTP URIs, and are represented in HTML (HyperText Markup Language). Furthermore, the HTML documents also include hyperlinks to other documents, making the process of discovering new documents easier for the users [4]. In linked data, instead of links between the documents, there are data-level links that connect data from different sources into a single global data space [10].

2.3 Linked Open Data

Linked Open Data is a concept proposed by Berners-Lee in 2010 [11] where linked data is available for everyone to use and reuse for free. It proposes five principles that aim to define standards and increase the interoperability of data. These principles follow a five-star rating system, where the more stars a dataset has, the more powerful and easy to use it is. The principles are as follows [11]:

1. Available on the web (whatever format) but with an open licence, to be Open Data
2. Available as machine-readable structured data (e.g. Excel instead of image scan of a table)
3. Available as machine-readable structured data plus non-proprietary format (e.g. CSV instead of Excel)
4. All the above plus, use open standards from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff
5. All the above, plus: link your data to other people's data to provide context

While the first four principles are generally followed in most linked data applications and datasets, the last principle can sometimes be neglected. When a dataset does not include links to other ones available on the web, there are many lost opportunities. Firstly, when the links are present, it is possible to discover new related data, increasing access to information. Furthermore, the links make the data present on a dataset discoverable, increasing its value. However, linking the dataset to external ones needs an investment of resources, as this is generally a manual task [5]. This research aims to design solutions to make it possible for linked data applications to connect and query external datasets, taking advantage of that data, and exploring in particular issues related

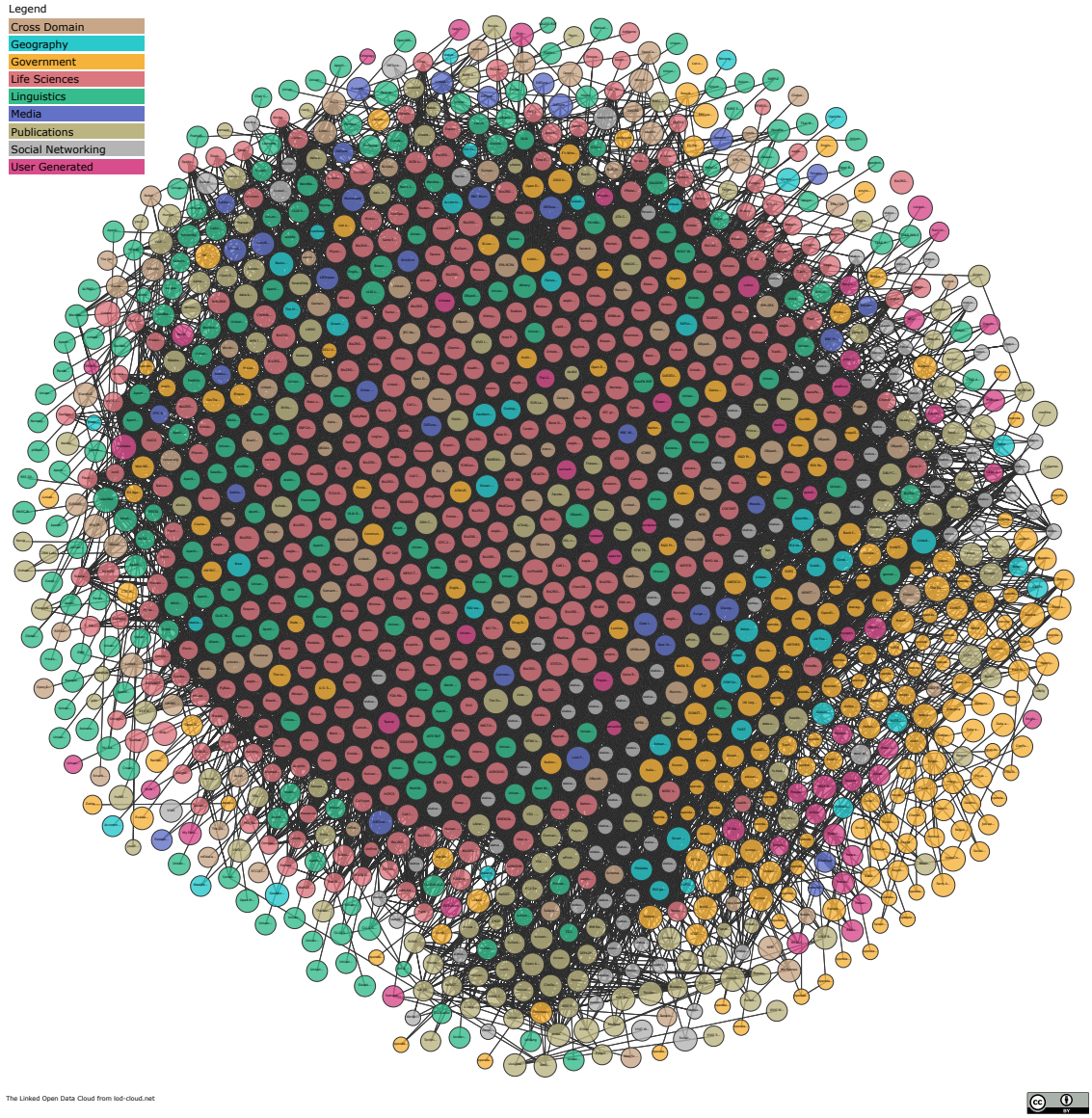

Figure 2.1: The Linked Open Data Cloud in 2022-11-03 [12].

to user interaction, so that the number of linked data applications and datasets classified with five stars under the Linked Open Data ranking system can increase.

The Linked Open Data Cloud, visible in Figure 2.1, is a diagram intended to provide a visual representation of the extent and variety of linked data that is currently available on the web, showing open datasets and the relationships between them [12]. This diagram demonstrates the huge amounts of data available for anyone to use and reuse, from a wide variety of fields and domains, such as geography, government, life sciences, linguistics, media, publications, social networking, cross-domain, and user-generated. One of the goals of this work is to take advantage of these datasets in the context of a linked data application.

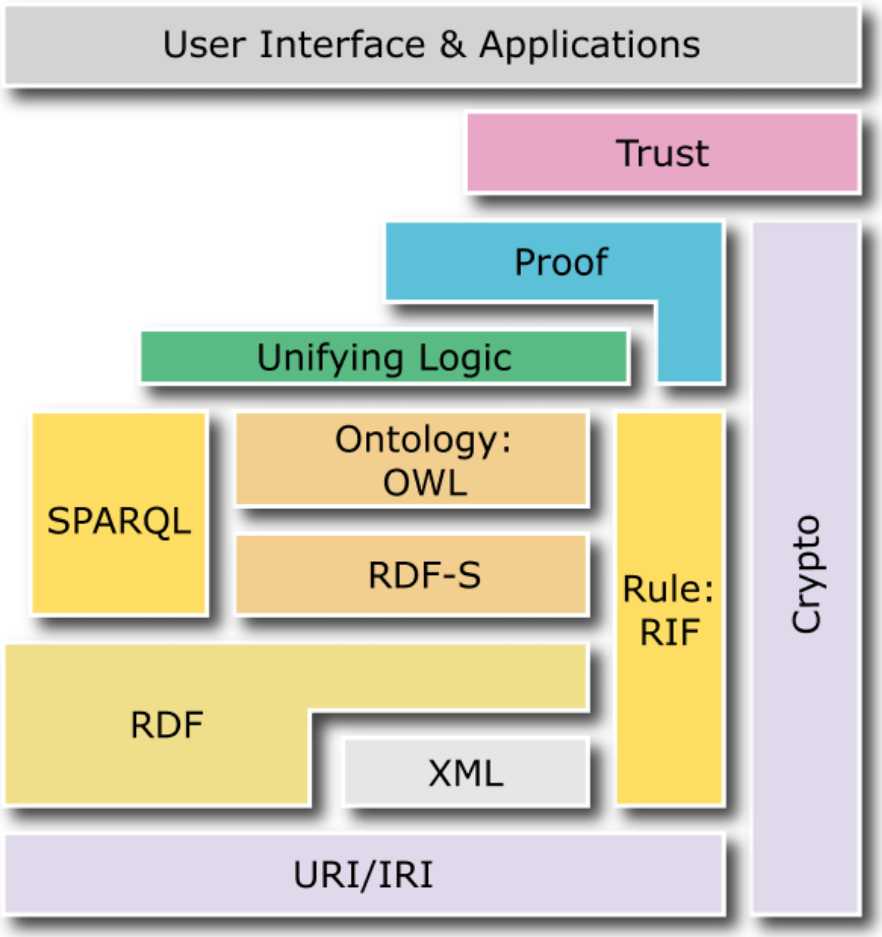

Figure 2.2: The semantic web stack [13].

2.4 Standards and Technologies

Figure 2.2 presents the semantic web technological stack, presenting some of its standards and technologies. Each layer has a specific function and interacts with other layers [13].

The bottom layer, the URI/IRI, provides a way to unambiguously identify objects in the semantic web. Uniform Resource Identifiers (URIs) and the HyperText Transfer Protocol (HTTP) are two fundamental technologies to the web, being also the base of the semantic web. URIs are used to identify any entity that exists in the world and can be dereferenced over the HTTP protocol [4]. In this way, when a user looks up the URI of an entity, more information about that entity can be retrieved.

Other critical layers are RDF and XML. Resource Description Framework (RDF) is a generic graph-based data model that provides a way to express and link data. RDF uses an XML-based syntax and supports the use of XML schema datatypes [14]. In RDF, data is expressed in triples with the form *subject, predicate, object*. The subject is a resource identified by an URI and the object can be another resource or a literal. The predicate, sometimes referred to as the property, is

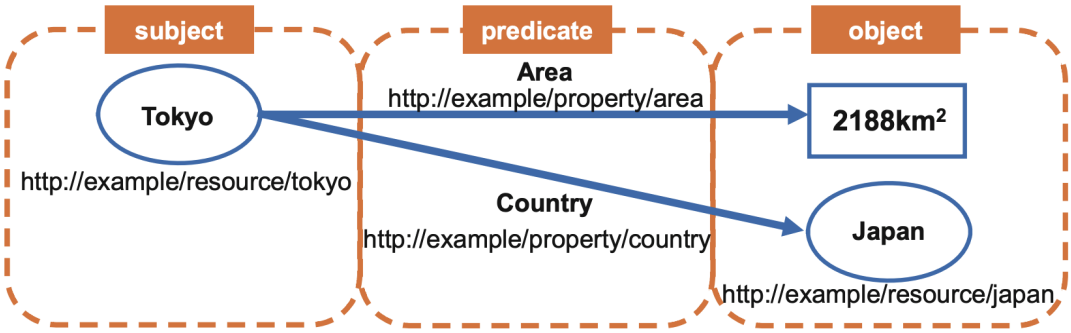

Figure 2.3: Data represented in the RDF data model [15].

also represented by an URI, and denotes the relationship between the subject and the object [4]. Figure 2.3 presents an example of data represented in the RDF data model, where the resources are represented by an oval, the literal by a rectangle and the predicates by a directed arrow. The two represented facts are that Tokyo has an area of 2188km^2 and Tokyo is located in the country Japan [15].

The RDF Vocabulary Definition Language (RDFS) [16] and the Web Ontology Language (OWL) [17] layers are used to create vocabularies that describe entities and how they are related [4]. OWL is more powerful than RDFS, as it provides a richer set of constructs for representing knowledge, such as cardinality and value constraints, class axioms, and individual identity. Furthermore, OWL provides formal reasoning, meaning that inferences can be automatically made based on the ontology, while RDFS provides only basic inferencing capabilities [17].

Finally, SPARQL Protocol and RDF Query Language (SPARQL) is a query language for RDF. This language can be used to express queries for retrieving and manipulating RDF data, and has many advanced capabilities, such as aggregation and subqueries [18].

2.5 Visualization Techniques

The existence of large quantities of data on the semantic web brought the necessity of new techniques for the visualization of linked data. The goal of these techniques is to provide suitable graphical representations of the information within a dataset, depending on the type of data and the task the user is trying to perform [9].

Figure 2.4 illustrates the linked data visualization process [9]. The first step is to extract the relevant data from the dataset using a SPARQL query. Next, if necessary, the data must be transformed so it can be presented using the desired visualization techniques. Finally, the data must be mapped to the components of the visualization. The resulting view may not be a static image and could allow user interaction, like zooming or clicking to provide additional visualizations [9].

Figure 2.5 presents an example of the linked data visualization procedure [9]. Firstly, the data of interest, in this case the number of Beatles albums released in each country, is obtained

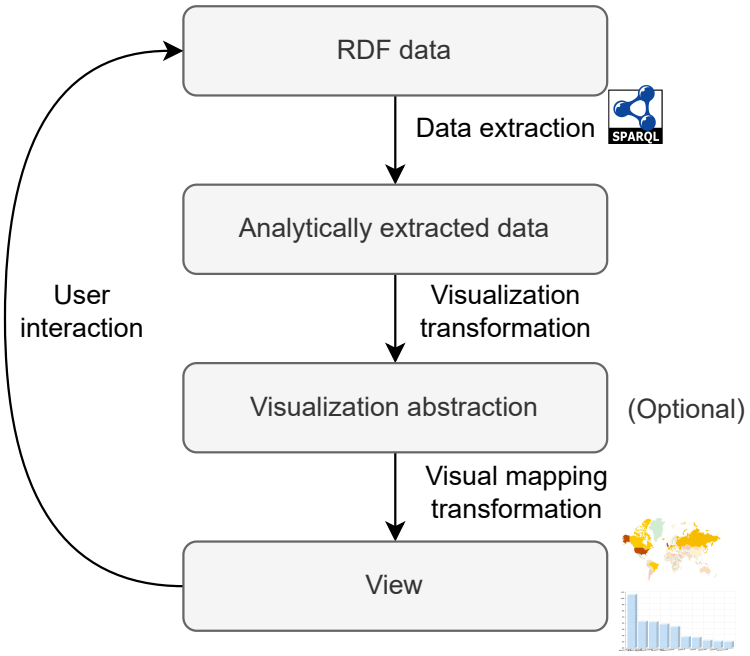

Figure 2.4: Linked data visualization process [9].

using a SPARQL query. Next, the country name is transformed into a country code. Finally, the data is displayed on a heatmap, represented in Figure 2.6, where the country code locates the corresponding area on the map and the number of releases determines the intensity of color in that region [9].

Due to the nature of linked data, there are still a number of challenges in linked data visualization [9]. Firstly, achieving scalability can be very difficult, as the amount of data is generally very high, which can bring performance problems. One way to make linked data applications more scalable is to provide flexible user interaction, as it is not necessary to provide static views with all the data of interest and users should be able to navigate and explore the data. Moreover, there is a challenge of creating generic visualization tools that can be used with many datasets, such as maps and timelines. Finally, the data may be partitioned across different repositories, which introduces data management complexity. The data may be modeled in different ways and have different formats across the datasets. Furthermore, the likelihood of missing data increases with the number of different repositories [9].

2.6 Challenges

Despite all the benefits of linked data, there are still some challenges that require further research to be surpassed. Firstly, ensuring data quality is a challenge, because everyone can publish data on the web. Hence, there is a need to trust the data that is being used. This issue is also present on the traditional web, where the quality and trustworthiness of the documents are difficult to ensure. Link analysis algorithms, such as PageRank [19] or HITS [20] can contribute to the reduction of

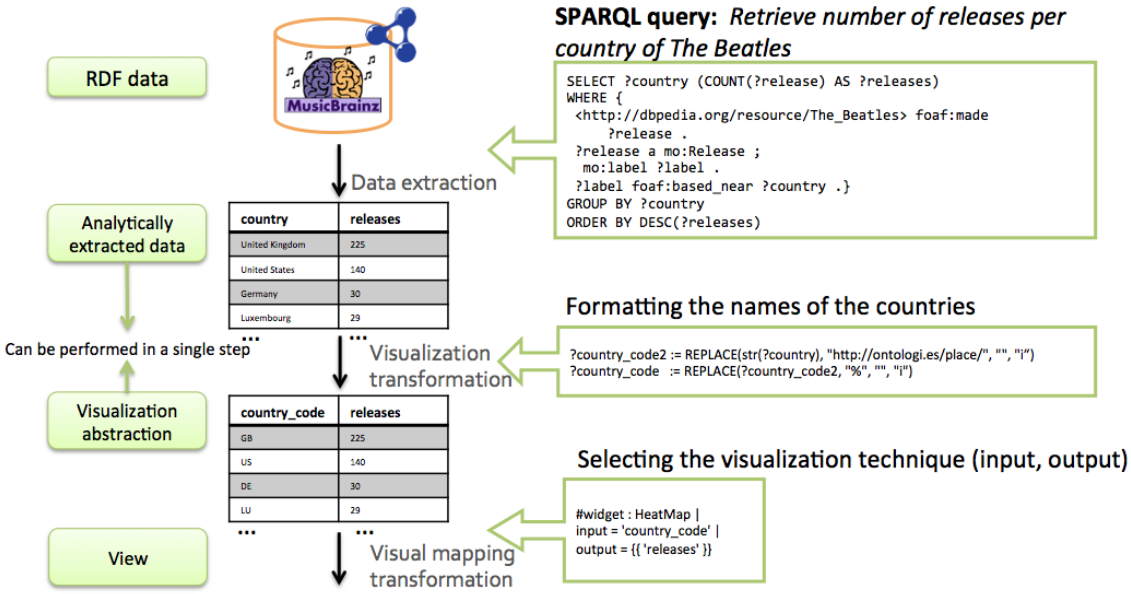

Figure 2.5: Example of the linked data visualization process [9].

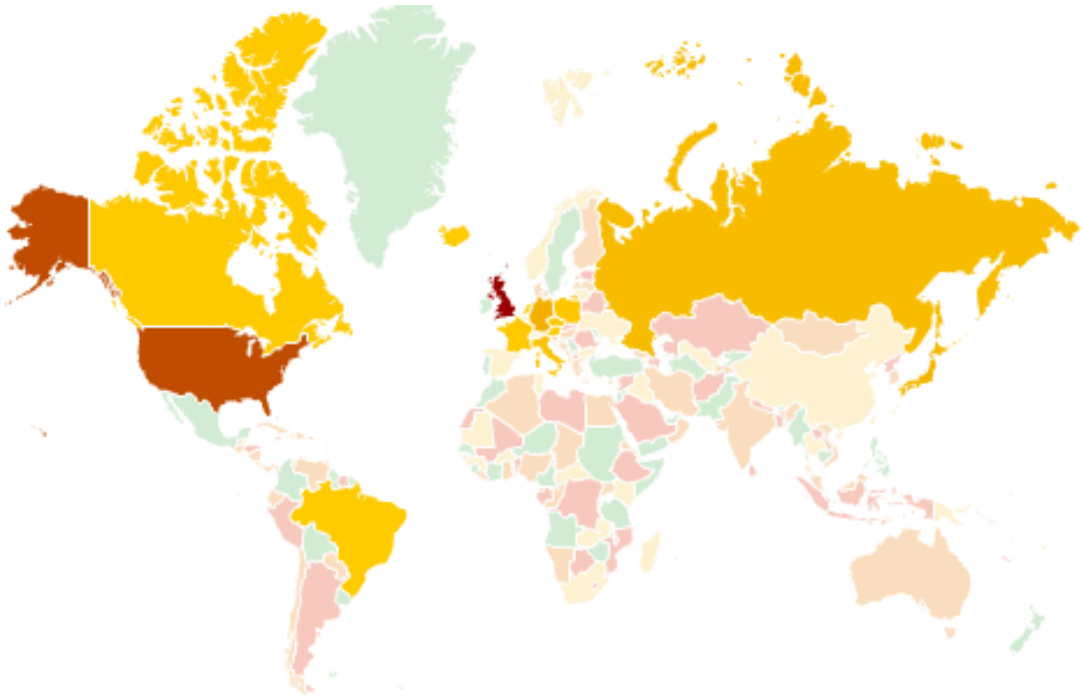

Figure 2.6: Heatmap visualization of Beatles releases [9].

this problem in the context of web search, by improving the search results of a user query. Similar approaches were also studied for linked data with the goal of improving the quality of the results when a user is querying a dataset, such as exploiting semantic tagging relying on Linked Open Data [21].

As in the case of the web of documents, managing the huge quantities of open data on the semantic web is a complex task. Furthermore, making sure the data is used in an ethical and responsible manner is a difficult task [22].

One additional challenge of using linked data is the technical expertise that is needed to work with it. Because the semantic web is still not as widely used as the traditional web, it is difficult to find developers with the necessary knowledge to develop linked data applications. Moreover, the semantic web technologies still do not have the desired maturity level. This challenge should be surpassed as the number of open-source linked data applications increases.

Finally, integrating data from different sources is one of the biggest challenges of linked data, mainly because linking data from different datasets and reconciling differences in the data models and vocabularies used is not straightforward, often even requiring manual work. Despite the existence of a large number of datasets published on the web of data, the access to information tends to be focused on a specific dataset, neglecting the full power of data integration and interoperability [23]. One of the goals of this research is to help surpass this challenge, especially by exploring the SPARQL query federation technique.

2.7 Linked Data Datasets and Applications

Several linked data datasets and applications have been developed to take advantage of the numerous volumes of linked data available on the web. In this section, we will start by presenting the two most known linked data datasets, DBpedia and Wikidata (Sebsection 2.7.1). According to Christian Bizer, Tom Heath, and Tim Berners-Lee, there are three categories of linked data applications: linked data browsers, linked data search engines, and domain-specific linked data applications [4]. Linked data browsers allow users to navigate between the web of data by following links expressed as RDF triples. On the other hand, linked data search engines aim to offer human users keyword-based search services [4]. Due to the nature of this research, we will then focus on the domain-specific linked data applications, which are developed taking into account a specific domain (Subsection 2.7.2). We will present four examples of domain-specific linked data applications: ResearchSpace, Arches, DOREMUS, and Revyu.

2.7.1 Linked Data Datasets

DBpedia is one of the most known knowledge base on the semantic web. It is a community effort to extract information from various Wikimedia projects, such as Wikipedia, and make it available on the web as linked data. DBpedia's knowledge base is interlinked with other open datasets on the semantic web, serving as a nucleus for the web of data [24].

Table 2.1: Categorisation of semantic web applications [26].

Dimensions	Levels	Description
Semantic technology depth	Extrinsic	Use of semantics on the surface of the application.
	Intrinsic	Conventional technologies (e.g., RDBMS) are complemented or replaced with SW equivalents.
Information flow direction	Consuming	LD is retrieved from the source or via a wrapper.
	Producing	Publishes LD (in RDF-based formats).
Semantic richness	Shallow	Simple taxonomies, use of RDF or RDFS.
	Strong	High level representation formalisms (OWL variants).
Semantic integration	Isolated	Creation of own vocabularies.
	Integrated	Reuse of information at schema or instance level.

Wikidata is a project that aims to create new ways for multiple Wikimedia projects to manage their data using linked data principles. It provides an open knowledge base to which everyone can contribute [25]. As in the case of DBpedia, it is connected to other datasets and is one of the most important knowledge bases on the semantic web, being the source of information for multiple linked data applications.

2.7.2 Linked Data Applications

A typical linked data application has three main components: the linked data consumer, the linked data manager, and the user interface [26]. The linked data consumer is responsible for acquiring linked data from various sources and converting non-RDF data into linked data, if necessary. The linked data manager manipulates the consumed data and can generate new linked data. Finally, the user interface provides a means of interacting with the application [26].

In addition to the categories of linked data applications discussed at the beginning of this section (linked data browsers, linked data search engines, and domain-specific linked data applications), linked data applications can also be categorized based on technical aspects that describe how the linked data is represented and utilized [26, 27]. These categories are listed in Table 2.1.

Firstly, with respect to the semantic technology depth, an application can be extrinsic, intrinsic, or both. When the semantic technology is extrinsically used, the linked data is accessed and processed through APIs. However, the application uses traditional systems like Relational Database Management Systems (RDBMS) for data management. On the other hand, the semantic technology depth is classified as intrinsic when conventional technologies are complemented or replaced with semantic web equivalents. One example of an intrinsic use of semantic technology is when an application uses a triplestore for storing its internal state [26].

Linked data applications can also be categorized in the dimension of the information flow direction. An application can consume linked data, produce linked data, or both [26]. Regarding the semantic richness, applications can be classified as shallow when using simple taxonomies, such as RDF or RDFS, or as strong when using high-level representation formalisms, such as OWL [26].

Finally, concerning the semantic integration, a linked data application can be isolated when using its own vocabularies, or integrated, when it reuses and interlinks vocabularies already available on the semantic web. Linked data applications should be integrated to support a more connected and cohesive web of data, allowing for more effective information exchange [26].

2.7.2.1 Examples of Linked Data Applications

ResearchSpace [28] is an open-source project designed at the British Museum that aims to provide a set of RDF datasets and tools to describe concepts and objects related to cultural historical research. The interactive tools allow users to access the data and contribute to the dataset by creating RDF annotations [26].

Arches [29] is an open-source platform developed by the Getty Conservation Institute and the World Monuments Fund for managing cultural heritage data. The platform is used by a variety of heritage institutions worldwide. The system's goal is to manage inventories of cultural heritage places, such as monuments [30].

DOREMUS [31] is a project with the goal of creating tools and techniques for describing, publishing, connecting, and providing context to musical catalogs on the semantic web. Overture [32] is a linked data application developed by the DOREMUS team to allow the public to explore its interlinked catalogs [33].

Revyu is a reviewing and rating website that uses semantic web technology and consumes and publishes linked data. The data is linked to external sources such as DBPedia to enrich the reviews and ratings with additional related information [34].

In conclusion, the projects presented above are just examples of the use cases linked data can provide. The large amount of structured information available in the web of data opens opportunities for the appearance of new domain-specific linked data applications. Moreover, linked data applications can publish new data, enriching the Linked Open Data Cloud, and interact to bring more powerful functionalities.

Chapter 3

Federation in Linked Data

Data integration, and in particular, federation, is a fundamental concept in the Semantic Web, allowing the web of data to reach its full potential. Federation allows for querying multiple sources simultaneously, bringing the opportunity of creating more complex queries. In this chapter, we present a starting point for a state of the art survey of federation in linked data. We start with an overview of the different phases of interaction with the Web of Data. This section is divided into Subsection 3.1.1, presenting the steps for publishing linked data on the web, and Subsection 3.1.2, where linked data integration techniques are introduced. Section 3.2 offers an overview on SPARQL endpoint federation systems. Finally, Section 3.3 presents examples of linked data applications that use the federation technique.

3.1 Interacting with the web of data

Given the large quantities of linked data available on the semantic web, it is important to have mechanisms to interact with it. More specifically, publishers need to expose their data, and consumers need to retrieve it.

3.1.1 Publishing Linked Data

After the creation and interlinking of a linked data dataset, the following steps should be followed for the completion of the publishing process [35]:

1. **Creation of metadata for describing the dataset**. This step involves the specification of additional information about the dataset, such as its name, description, creator, creation data, example resources, and where it will be available. The VoID (Vocabulary of Interlinked Datasets) [36] is generally used for describing RDF datasets. The correct and complete description of the dataset is a fundamental step for federation, as we will see in Section 3.2.

2. **Making the dataset accessible**. This can be done using four different mechanisms [35]:

   (a) **Dereferencing HTTP URIs**, by looking up the resolution of an HTTP URI, as stated by the second linked data principle (Section 2.2).

   (b) **RDFa**, when RDF is embedded in HTML documents.

   (c) **SPARQL Endpoint**, a service identified by a URL, that uses the SPARQL protocol for SPARQL queries and returning the results.

   (d) **RDF dump**, when the RDF dataset is contained in files that can have a different number of formats, such as RDF/XML or N-Triples.

3. **Exposing the metadata in linked data repositories**. Publish the dataset metadata in linked data repositories, such as CKAN [37] and Data Hub [38], so that other people can discover it.

4. **Validating the dataset**. Perform an RDF dataset validation, with steps such as checking if URIs are dereferenced correctly and checking the RDF syntax.

3.1.2 Querying Linked Data - Linked Data Integration

The simplest way to query the web of data is by executing SPARQL queries on a specific dataset, through its SPARQL endpoint. However, this approach does not take into account the links between the datasets. To fully take advantage of the web of data, there is a need for a virtual union of linked data from multiple datasets for executing queries [39]. There are two paradigms for querying distributed data on the semantic web: data warehousing and federated query processing [39].

In the data warehousing approach, illustrated in Figure 3.1, data is collected and stored in a central repository, and the query is executed from that dataset. In this approach, queries are executed almost instantaneously. However, this advantage comes with the cost of storage and maintaining the repository updated [40].

On the other hand, in the federated query processing approach, depicted in Figure 3.2, the query is distributed over the SPARQL endpoints associated with each dataset [39]. A mediator is used to transform the query into several sub queries, executed in each dataset, and integrate the respective sub results into a final result [40]. Contrary to the data warehousing approach, there is no need to maintain an updated data repository, but the query processing takes longer [40]. Furthermore, this approach allows for data integration scalability and is easier to implement. Federated query processing can be performed natively using SPARQL 1.1 in any query service on the web, making this approach more desirable when compared to data warehousing. Therefore, we will focus on the federated query processing approach, which will be presented in more detail in the next section (Section 3.2).

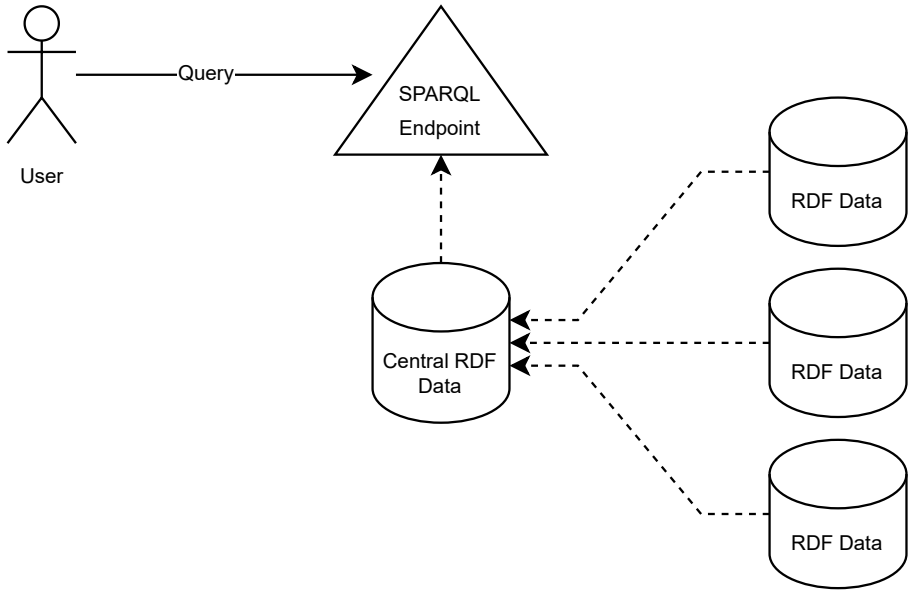

Figure 3.1: Querying distributed data using the data warehousing approach [40].

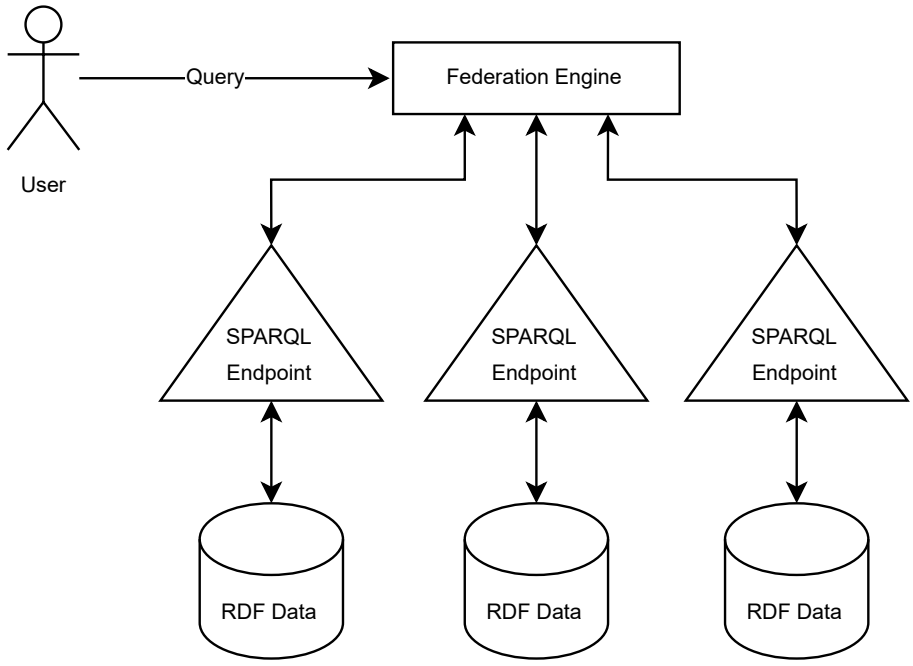

Figure 3.2: Querying distributed data using the federated query processing approach [40].

For querying data from multiple sources, it is essential that the datasets to be queried are linked. The linking can be done by entity matching, which is the process of connecting two related entities from different datasets [40]. Entity matching is generally done by generating triples with the `owl:sameAs` property, which states that the subject and the object are the same entity [40].

3.2 SPARQL Endpoint Federation Systems

Version 1.1 of SPARQL enables the execution of federated queries natively [41], by using the `SERVICE` keyword for specifying the SPARQL endpoint of the desired dataset [39]. Listing 3.1 presents an example of a federated query, where a remote SPARQL endpoint is queried to retrieve the name of a person.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX I: <http://example.org/I>

SELECT ?name
WHERE {
  I: foaf:knows ?person .
  SERVICE <http://people.example.org/sparql> {
    ?person foaf:name ?name .
  }
}
```

Listing 3.1: SPARQL federated query

Despite the possibility of executing federated queries natively in SPARQL 1.1, there is a more powerful approach to query multiple linked data sources, by using a SPARQL endpoint federation system, also known as SPARQL query federation system or distributed query processing.

A SPARQL query federation system provides access to multiple datasets from different SPARQL endpoints and uses a combination of data federation and query optimization to answer user queries [42]. Figure 3.3 presents the steps for distributed query processing [7]. The first step is to parse the query and transform it from SPARQL to a query execution tree. Secondly, the system performs the dataset selection, or SPARQL endpoint selection, for each triple. Next, the query is optimized to improve its performance, by placing the triple patterns into groups and rearranging the order of joins and triple patterns. Finally, the query is executed.

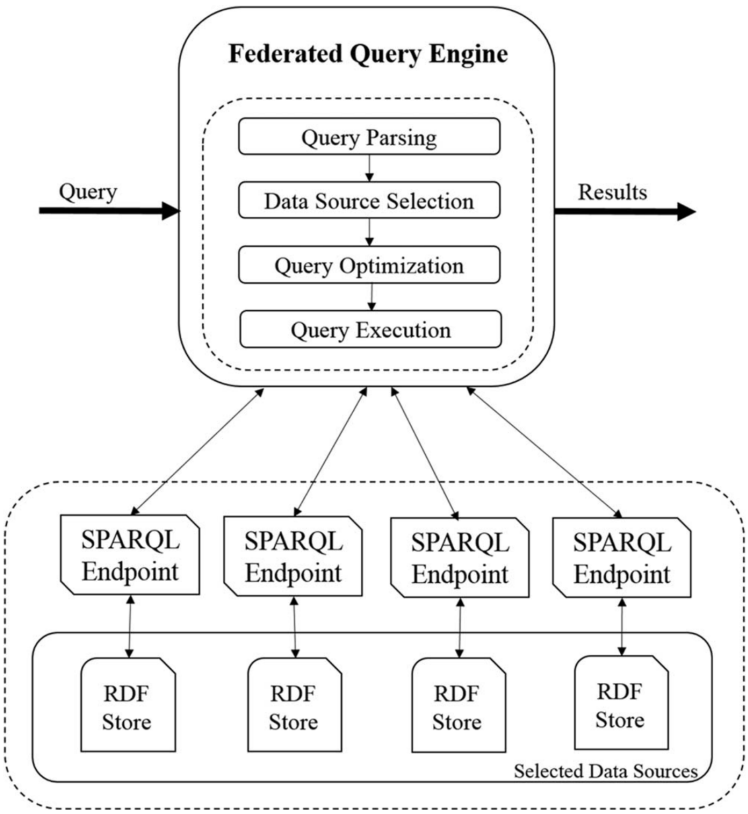

Figure 3.3: Federated Query Engine [43].

Using SPARQL endpoint federation systems has many advantages over raw SPARQL federated queries.

Firstly, these systems perform distributed query optimization to minimize data transfer, minimize latency and enhance performance, reducing the time it takes to execute the queries. Depending on the complexity of the federated query, the performance gains can be very significant [8].

Another advantage of using federated query processing is the dynamic SPARQL endpoint discovery. While in SPARQL there is a need to specify the SPARQL endpoints, using the `SERVICE` keyword, federation systems can automatically discover the SPARQL endpoints associated with the query. There are multiple approaches for dynamic SPARQL endpoint discovery. One example is querying central registries of datasets metadata, such as the Linked Data Fragments registry [44]. Another approach is to let the user provide a set of endpoints, and the system will automatically decide when to use which one [8]. The source selection can be done using the `ASK` SPARQL query. This query returns a boolean value stating if the query can be answered by the specified SPARQL endpoint. In this way, the federation system can decide the SPARQL endpoints associated with each sub queries [40]. Most federation systems use caching mechanisms to reduce the bandwidth consumption of the additional `ASK` SPARQL queries and improve performance [40].

There are several surveys on SPARQL endpoint federation systems, comparing the different

approaches for the distributed query processing steps [8, 45, 43, 40]. In these surveys, one of the engines that stands out the most is FedX [46], a federation engine developed on top of the RDF4J framework [47].

3.2.1 Challenges

Despite all the benefits of using federation systems, its usage is more complex when compared to the raw SPARQL query approach. Moreover, there are still some challenges and open research issues in this field [43, 40].

The first challenge is metadata management. For the data source selection step, it is essential to access central repositories of metadata about open semantic datasets. The VoID vocabulary [36] aims to be a standard for RDF dataset descriptions, as it provides metadata of data sources and their relations. However, there are still some open research questions about how to manage these repositories, such as how to generate the metadata and how often the repository should be updated [43]. Furthermore, not all semantic datasets are present in these catalogs, which makes them invisible to the federation system.

The caching of the results is also a challenge in federated query processing. Although multiple studies were made about the caching of results in diverse contexts [48, 49, 50, 51, 52, 53], how to effectively use non-naive caching mechanisms in query federation engines is still an open question [43].

Lastly, the overlapping terminologies for the same concepts can be an issue for distributed query processing [40]. The web of data contains large amounts of linked data, which leads to the existence of repeated information, represented using numerous vocabularies and terminologies. Consequently, there is a need for the usage of entity mapping for distributed query processing. In other words, the data sources should map related concepts and generate more links among related entities to fully take advantage of federation [40].

3.3 Linked Data Applications using Federation

With the increasing number of linked data applications, it is important that federation is used to fully take advantage of the semantic web. This section presents some linked data applications that use the federation technique.

The Information Workbench [54] is a platform for the development of linked data applications. It abstracts the technical details behind the application development and deployment and allows for data management and user interface customization. The platform supports the integration of data from multiple sources using the FedX SPARQL federation system [54]. Figure 3.4 presents the architecture of the federation layer in Information Workbench. The FedX engine distributes a query through multiple SPARQL endpoints associated with each data source. Furthermore, it uses metadata registries such as CKAN [37] and data.gov [55].

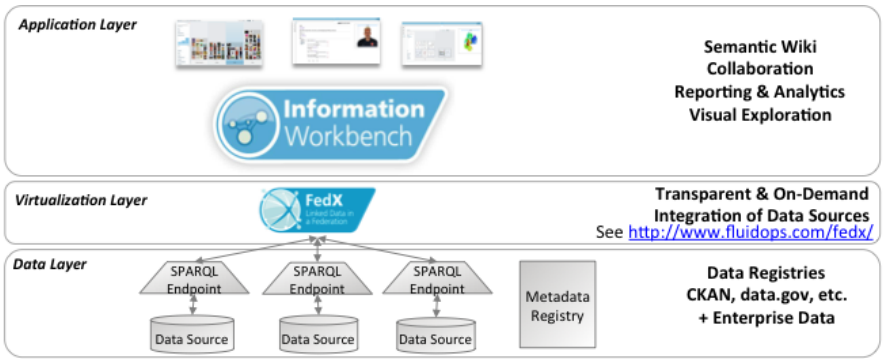

Figure 3.4: Federation using FedX in Information Workbench [26].

LOD4Culture (Linked Open Data for Culture) [56] is a project that aims to create a linked open data platform for cultural heritage institutions in Europe. It has an interactive map for exploring cultural heritage sites worldwide, and combines data from Wikidata and DBpedia to expand the available information [56]. Figure 3.5 presents two screenshots of the application, namely the interactive map, and a page with information about a monument. LOD4Culture uses federation through the CRAFTS API [57], which translates REST API calls into SPARQL queries that are sent to multiple data sources.

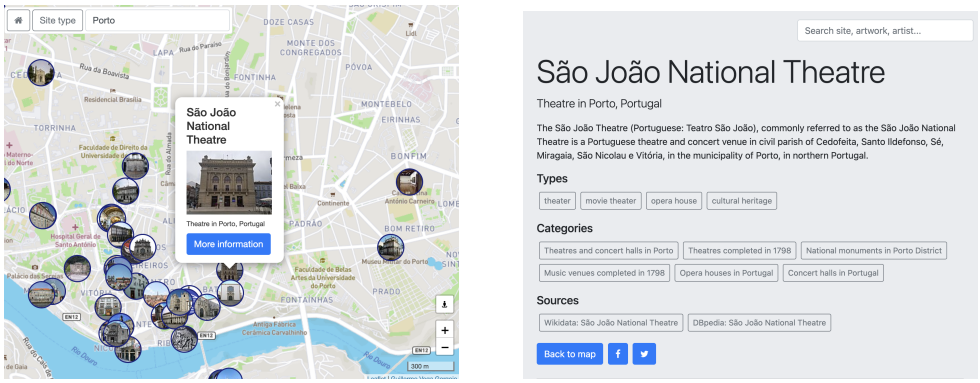

Figure 3.5: LOD4Culture Platform.

Chapter 4

Federation Solutions for Linked Data Applications

This chapter presents the research problem that our work aims to solve (Section 4.1) and the respective proposed solution (Section 4.4). Furthermore, it introduces the domain and the project that will be used as a case study for the proposed solution, namely the archival domain (Section 4.3) and the EPISA project (Section 4.2).

4.1 Problem

The research problem of this work can be defined by the following research questions:

1. How to integrate data available on the semantic web on linked data applications using federation?
2. How to provide an intuitive user interface for the discovery, integration, and exploration of new data in linked data applications?

These research questions were derived from the following hypothesis:

> Using federation to integrate data available on the semantic web can substantially enhance a linked data application by giving its users access to more useful and detailed information according to the usage context.

While the idea behind this hypothesis makes sense in theory, putting it into practice is not trivial. This research aims to tackle the technical difficulties involved in these integration techniques.

4.2 EPISA Project

The EPISA project, developed in the archival domain, will be used as a case study for the proposed solution. This section presents the EPISA project and how it could be improved using linked data federation.

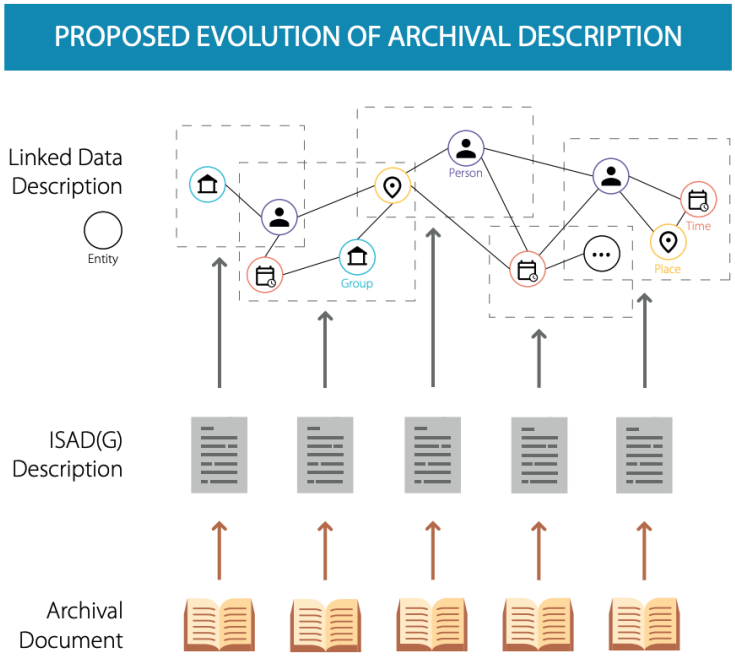

Figure 4.1: EPISA proposed evolution of archival records [60].

The EPISA project (Entity and Property Inference for Semantic Archives) [58] was developed in the context of the ongoing data infrastructure renewal in DGLAB (General Directorate for Book, Archives and Libraries), the institution responsible for the Portuguese National Archives. The goal is to develop a prototype for an open-source platform adopting a new data model for archival description based on linked data principles [59].

Figure 4.1 presents the proposed evolution of archival description. From the bottom up, we can see that the physical archival records are described to the ISAD(G) model, and then migrated to a description model based on linked data principles [60].

Figure 4.2 showcases a concrete example of an archival record described using the ISAD(G) description model and how it is migrated to linked data, contributing to the atomization of the data [59]. The scope and content ISAD(G) field, highlighted on the left side of the figure, is converted from a text representation to a graph, making the implicit entities and relationships between them explicit [60].

The work developed in the EPISA project can be used to facilitate the access to the archival information and give users a better understanding of the repository [59].

4.3 Linked Data in the Archival Domain

Archives are contemporary records created by individuals and organizations to register information about past events. These records can have a wide range of formats, including written, photographic, and sound, and can be either digital or analog. Archival records are kept by public and

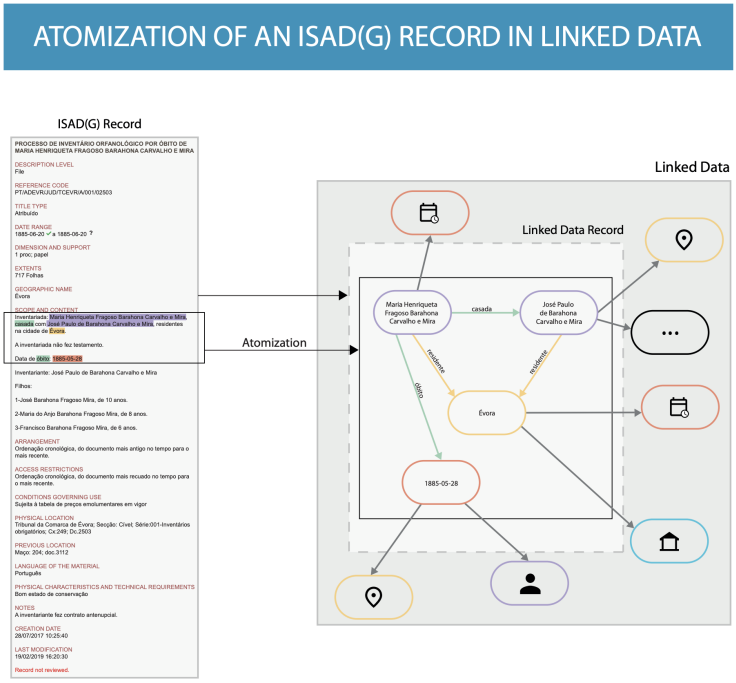

Figure 4.2: EPISA atomization of an ISAD(G) record in linked data [60].

private institutions [61].

Because of the diversity of the archival universe, archival description is a fundamental task. The goal of archival description is to identify and explain the context and content of archival material to promote its accessibility [62]. In other words, archival description provides metadata about the archival records, allowing for its organization in accordance with predetermined models [62].

With the evolution of technology, and in particular, the web, the way archival description is accomplished has also evolved to incorporate new technologies to improve the search, retrieval, and use of archival materials [63]. More recently, there has been an effort to bring linked data principles to the archival domain, because of its multiple benefits [63]. Firstly, publishing archival data as linked data makes it machine-readable, which allows the use of automated processes for reasoning and analysis. Linked data machine readability and its ability to connect data from other datasets improves the user experience, as it allows for more meaningful archival descriptions and gives the user the possibility of further exploration [64]. Furthermore, linked data increases the access and ease of use of archival data, by increasing its openness, visibility, and discoverability, enabling archives to reach new and larger audiences [64].

4.4 Proposed Solution

To answer the hypothesis formed in Section 4.1, the solution will be a system that can be integrated into linked data applications to expand its knowledge bases using data integration techniques, namely SPARQL federation. The developed system will be integrated into the EPISA platform,

which will be used as a case study. Furthermore, a simple linked data application, Linked Classical, will be created, due to the complexity of the EPISA project. By developing Linked Classical, we can publish it as an open-source project, serving as a template for other linked data applications that want to use federation techniques.

Firstly, the system will identify resources that can be expanded with additional information from external knowledge bases. The second step will be to locate these resources in the Linked Open Data Cloud, retrieve their URIs, and generate new RDF triples that establish links between the resources from the application's knowledge base and external sources. Then, additional information about the resources will be retrieved from the semantic web, using federation. The development of the system will have an emphasis on user interaction. Thus, linked data visualization techniques will be explored to enhance the understanding of the expanded resources.

The system integrated into the Linked Classical platform will go through a functional evaluation process, for the validation of the requirements of this application.

Chapter 5

Linked Classical Application

This chapter presents and describes the approach used to address the problem of how to integrate data available on the semantic web on linked data applications using federation, and how to provide an intuitive user interface for the discovery, integration, and exploration of new data in linked data applications. The primary objective is to introduce Linked Classical, a linked data application developed within the context of this work, serving as an example for the future development of linked data applications utilizing federation. The chapter describes the design decisions made during the implementation of the Linked Classical application, highlighting the strategies and considerations employed to create a linked data application that effectively leverages federation.

5.1 The Linked Classical Application

The emergence of the semantic web has unveiled new possibilities and applications that capitalize on the interconnected nature of linked data. At the core of the semantic web are linked data technologies, enabling the representation, integration, and utilization of structured data in a machine-readable format. Despite the considerable potential and opportunities offered by the semantic web, its widespread adoption faces numerous challenges.

One of the primary obstacles is the scarcity of open-source projects and proficient developers who possess an in-depth understanding and mastery of linked data technologies. These technologies, given their relative novelty and inherent complexity, have limited the number of experts in the field. Consequently, the availability of well-established open-source projects that effectively employ linked data technologies remains limited. This scarcity of resources delays developers and researchers from readily exploiting cutting-edge linked data technologies to construct innovative applications.

The EPISA project's inception highlighted this challenge, as the team encountered difficulties in locating relevant open-source projects that utilized linked data technologies. Consequently, a considerable amount of time and effort was expended in setting up the necessary technologies and establishing an appropriate development environment. This experience underscores the need

for comprehensive resources, well-documented libraries, and accessible tools that facilitate the development of linked data applications.

To address these challenges and provide guidance for the future development of linked data applications, we introduce the Linked Classical application. Developed within the context of this dissertation, the Linked Classical application serves as a tangible and practical example, showcasing the potential and feasibility of linked data technologies in the domain of classical music. By demonstrating the implementation of linked data principles and data integration techniques, the Linked Classical application aims to inspire and guide future developers in their pursuit of creating their own linked data applications.

Through the exploration and development of the Linked Classical application, this work seeks to contribute to the advancement of linked data technologies and promote their wider adoption across various domains. By providing a tangible example and sharing valuable insights gained throughout the development process, this research aims to empower researchers and developers to leverage the semantic web and linked data technologies, thereby unlocking new possibilities in data integration, knowledge representation, and application development.

5.1.1 The Linked Classical Application as an Open-Source Project

The Linked Classical project is hosted on GitHub [65], a widely recognized platform for collaborative software development and version control. The Linked Classical GitHub repository, accessible at `https://github.com/feup-infolab/linked-classical`, stores the source code and documentation of the application.

In the root of the repository, there is a `README.md` file containing the documentation of the application. It includes information about the project, technical details such as the architecture and how SPARQL federation is employed, requirements and usage instructions, as well as a video demo.

The code is divided into two folders at the root of the repository:

- **frontend:** This folder contains the Laravel [66] code for the frontend, responsible for the user interface and interactions.
- **server:** Within this directory, we store the server-side components. This includes:
  - **fuseki:** This folder contains the essential code for configuring the Apache Jena Fuseki [67] triplestore, a critical part of data storage and retrieval.
  - **middleware:** This houses the Java Spring Boot [68] application, which acts as the REST API, ensuring seamless communication between the frontend and the triplestore.
  - **sources:** This directory stores the dbtune classical dataset as a turtle file. This dataset forms the foundation of the application's data.

The publication of the Linked Classical project is driven by a desire to offer a template for the development of similar linked data applications in the future. This open-source initiative is designed to assist developers and researchers in creating new linked data applications, thereby advancing the field of the semantic web.

5.2 Requirements

In this section, the requirements of the Linked Classical application are presented using user stories [69]. These user stories offer a valuable approach for capturing and communicating the functional requirements in a concise and user-centric manner. The main focus is on describing the desired features and functionalities from the perspective of the end user, ensuring that the application follows their specific needs and expectations. By employing user stories, the requirements can be expressed in a more understandable and relatable manner, making it easier for stakeholders to grasp the intended functionality and guiding developers in creating a user-centered application. Through the presentation of these user stories, this section aims to provide a comprehensive understanding of the requirements for the Linked Classical application, forming a solid foundation for its development and ensuring alignment with the desired user experience.

The user stories have been intentionally framed from the standpoint of a typical application user, without specific differentiation based on the various user types that the application might support. Although the application may serve diverse user roles, including general users and administrators with data modification privileges, we primarily concentrate on the fundamental functionalities that directly influence the end user experience. By adopting this approach, the complexity that arises from distinguishing between general users and administrator users, who have the capability to modify the data, is intentionally omitted. Furthermore, by allowing general users to actively contribute without restrictions based on roles, the application's knowledge base can grow naturally, creating a collaborative and inclusive environment for expanding knowledge.

Furthermore, the identified user stories seek to encompass a variety of interaction scenarios, with the intent of representing diverse perspectives on interaction and technological challenges. It is important to note that these user stories are not intended to provide a comprehensive or exhaustive coverage of all aspects within the scope of this specific application.

Table 5.1 presents the user stories of the Linked Classical application.

U01: As a user, I want to see the page of a generic resource so that I can see its outgoing and incoming relations

U01 emphasizes the need for a comprehensive page that presents the information related to a generic resource, that is not a composer or musical work, displaying its outgoing and incoming relations. This requirement enables users to understand the interconnections among data and explore the relationships between different resources.

Table 5.1: Linked Classical User Stories.

Code	User Story
U01	As a user, I want to see the page of a generic resource so that I can see its outgoing and incoming relations
U02	As a user, I want to see a custom page when the resource I'm viewing is a composer or a musical work so that I can see its information in a more user-friendly way.
U03	As a user, I want to click on a resource URI and navigate to its page so that I can see its information.
U04	As a user, I want to see a map with the locations where a composer has lived and more information about these locations so that I can learn more about the composer's past.
U05	As a user, I want to link external entities to the resources of the knowledge graph so that I can enrich the information presented to the users.
U06	As a user, I want to see a section on the resource page with information extracted from external sources so that I can have more context about the information.
U07	As a user, I want to see a section on the resource page with equivalent external entities so that I can look up those entities in other sources.
U08	As a user, I want to search for composers and musical works so that I can find relevant information about classical music.

U02: As a user, I want to see a custom page when the resource I'm viewing is a composer or a musical work so that I can see its information in a more user-friendly way

U02 highlights the importance of creating custom pages dedicated specifically for composer and musical work resources. These pages should offer a user-friendly representation of the information, presenting it in a structured and easily understandable manner. By designing custom pages, the Linked Classical application enhances the accessibility and readability of composer and musical work details, outlining the importance of these two types of resources in the Linked Classical context.

U03: As a user, I want to click on a resource URI and navigate to its page so that I can see its information

U03 emphasizes the necessity of enabling users to navigate to a resource's page by clicking on its URI. This functionality ensures a seamless browsing experience, allowing users to quickly access detailed information about specific resources of interest by taking advantage of the linked nature of the data. By implementing this feature, the Linked Classical application enhances user navigation and promotes efficient exploration of the knowledge graph.

U04: As a user, I want to see a map with the locations where a composer has lived and more information about these locations so that I can learn more about the composer's past

U04 addresses the requirement to present a map showcasing the locations where a composer resided, along with relevant information about these locations, such as its population. This feature enhances the users' understanding of the composers' historical context and facilitates a deeper exploration of their lives. By integrating geographic data and displaying it on a map, the Linked Classical application enriches the user experience by offering spatial perspectives and additional insights.

U05: As a user, I want to link external entities to the resources of the knowledge graph so that I can enrich the information presented to the users

U06 addresses the need to enable users to link external entities to the resources within the knowledge graph. This feature enhances the interoperability of the Linked Classical application by establishing connections to external datasets. Through the incorporation of external entities, the application enriches the available information for users, promoting a broader and more comprehensive perspective on classical music.

U06: As a user, I want to see a section on the resource page with information extracted from external sources so that I can have more context about the information

U07 highlights the need to display a section on the resource page that provides information extracted from external sources. This functionality augments the contextual understanding of resources by incorporating additional information from trusted external repositories, such as Wikidata and DBpedia. By presenting such information, the Linked Classical application enhances the comprehensive nature of the user experience, offering a broader context for resource exploration.

U07: As a user, I want to see a section on the resource page with equivalent external entities so that I can look up those entities in other sources

Finally, U08 emphasizes the requirement to present a section on the resource page containing equivalent entities from external datasets, such as Wikidata and DBpedia. This feature enables users to explore related information across various external sources, allowing for cross-referencing and facilitating further research. By providing access to equivalent entities, the Linked Classical application expands the scope of information available to users, promoting a comprehensive understanding of classical music resources.

U08: As a user, I want to search for composers and musical works so that I can find relevant information about classical music

Finally, U09 aims to highlight the importance of implementing a robust search functionality, allowing users to search composers and musical works. This feature enhances user accessibility and enables targeted information retrieval, supporting users in finding relevant information about classical music. By facilitating efficient searches, the Linked Classical application empowers users to explore the knowledge graph effectively.

5.3 Knowledge Sources

The Linked Classical application relies on the DBTune Classical [70] dataset, an extensive and interconnected dataset of classical music data. The DBTune project [71], driven by a mission to construct a structured and interconnected representation of music-related information following linked data principles, laid the foundation for the DBTune Classical dataset. To achieve this, the project meticulously collected data from authoritative sources, music archives, libraries, and cultural institutions devoted to preserving classical music heritage. This process involved careful data extraction to ensure the dataset's accuracy, consistency, and comprehensiveness.

While curating the DBTune Classical dataset, the DBTune project faced the challenge of reconciling and aligning data from various repositories to ensure a unified representation of shared entities. This critical step fostered coherent relationships within the dataset and facilitated seamless exploration of classical music knowledge. Furthermore, the DBTune project leveraged open data standards and used ontologies like Music Ontology (MO) to enhance the dataset's organization and interoperability. These strategic decisions enabled smooth integration with other linked data resources, such as DBpedia, DBTune/musicbrainz, and BBC/music, enriching the application's overall user experience.

As a result of the DBTune project's meticulous efforts, the DBTune Classical dataset emerged as a valuable resource. Researchers, developers, and classical music enthusiasts alike benefit from its wealth of information and interconnected structure. In the context of the Linked Classical application, this dataset plays a central role, empowering user-driven content creation, efficient navigation, and insightful exploration of classical music knowledge.

5.4 Architecture

The architecture of the Linked Classical application, illustrated in Figure 5.1, was designed to ensure efficient data management, seamless querying, and a user-friendly interface for users exploring classical music knowledge. The application operates within a docker environment, utilizing three interconnected containers: `fuseki`, `middleware`, and `frontend`. Each container serves distinct purposes, creating a robust and cohesive system.

The DBTune Classical dataset, a central component of the application, is acquired as an RDF file from the official DBTune Classical website [70]. When launching the middleware container for the first time, the RDF file is ingested into the application, forming the foundational knowledge graph.

The `middleware` container comprises two fundamental components. Firstly, the Java Spring Boot functions as the REST API, providing a standardized interface for seamless communication between the frontend and the backend. Accessible via port 8080, the REST API ensures efficient data retrieval and manipulation, facilitating smooth interaction with the dataset. Secondly, Apache Jena plays a pivotal role in the `middleware` container, acting as a bridge that connects the REST API to the underlying triplestore. Leveraging Apache Jena's capabilities, the `middleware` enables seamless data retrieval, updates, and processing of SPARQL queries, ensuring a streamlined flow of information between the frontend and the triplestore.

Within the `fuseki` container, Apache Jena Fuseki acts as the application's triplestore and SPARQL server. Exposing the application's SPARQL endpoint through port 3030, this container serves as the gateway for executing SPARQL queries on the dataset. The Fuseki UI, also accessible through port 3030, provides a web-based interface where users can interactively compose and execute SPARQL queries, being particularly valuable for testing purposes and accommodating the needs of advanced users seeking comprehensive dataset exploration.

On the frontend side, a user-friendly interface was developed using the Laravel open-source PHP web framework. Laravel offers extensive features and capabilities to build interactive and dynamic web applications, facilitating seamless navigation, data presentation, and user interaction. Accessible through port 8000, the frontend component offers users an intuitive and visually appealing experience.

Finally, DBpedia and Wikidata, with SPARQL endpoints available at `https://dbpedia.org/sparql/` and `https://query.wikidata.org/sparql`, respectively, are two crucial components when executing SPARQL federation, a feature elaborated in Section 5.8.

In summary, the architecture of the Linked Classical application maximizes the capabilities of each container, ensuring smooth integration between the frontend, middleware, and triplestore. This approach enables users to explore the DBTune Classical dataset effortlessly, discover interconnected classical music resources, and deepen their understanding and appreciation of the rich legacy of classical music.

5.5 Data modeling

Data modeling is a critical aspect of designing applications, especially for linked data applications like the Linked Classical application. Data modeling involves systematically organizing and structuring data entities and their relationships to create a coherent and standardized framework for representing information. In linked data applications, data modeling is essential for adhering to linked data principles, which promote data interconnection and interoperability across diverse datasets.

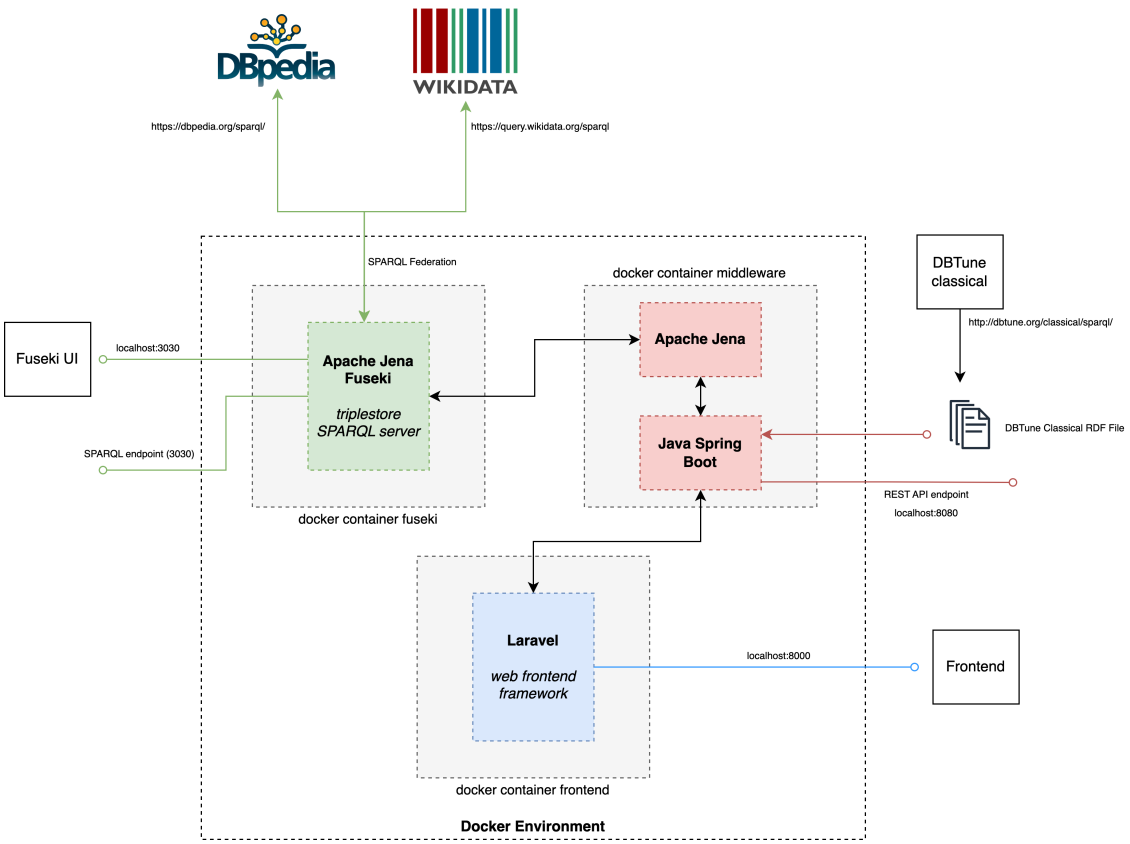

Figure 5.1: Linked Classical Architecture.

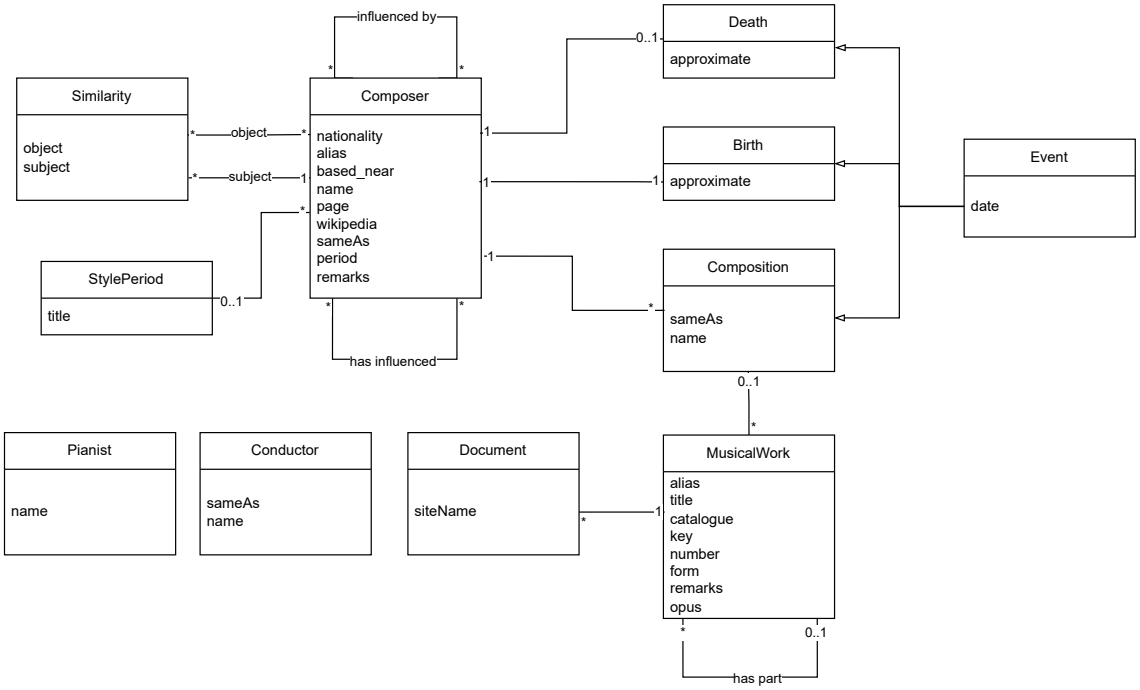

Figure 5.2: Linked Classical Class Diagram.

The primary objective of data modeling in linked data applications is to establish a semantic foundation that enables meaningful associations between data entities. By utilizing standard ontologies and vocabularies like RDF (Resource Description Framework) and OWL (Web Ontology Language), data modeling ensures data is expressed in a machine-readable format with uniform resource identifiers (URIs). This facilitates data integration, allowing the Linked Classical application to seamlessly link and integrate classical music data from different sources, enriching the collective knowledge graph.

Next, we will present the conceptual and logical data models. These models are derived from the already existing DBTune Classical dataset.

5.5.1 Conceptual model

Linked Classical conceptual model is represented in the class diagram of Figure 5.2. The class diagram provides a visual representation of the application's data model, showcasing various entity types and their relationships. The diagram highlights two main classes: Composer and MusicalWork, which form the core of the application, facilitating exploration and analysis of musical connections among composers and their compositions. Additionally, other classes include Similarity, StylePeriod, Death, Birth, Composition, Event, Document, Conductor, and Pianist.

The Composer class serves as a central entity in the application, representing composers of musical works. It has two many-to-many relationships with itself, indicating that a composer can influence and be influenced by other composers, creating a web of musical connections. To

facilitate exploration and discovery, there are two one-to-many relationships between Composer and Similarity, representing composers with similar styles. The Composer class is associated with a StylePeriod through a many-to-one relationship. This relationship allows a composer to be linked to a specific style period. Furthermore, the Composer class features one-to-one relationships with Death and Birth, symbolizing the respective events in a composer's life. Additionally, there exists a one-to-many relationship between Composer and Composition, which denotes that a composer can be associated with multiple compositions, with each composition exclusively linked to a single composer. The Event class acts as a parent class for Death, Birth, and Composition. It represents different events relevant to composers and musical works. The Event class allows for a unified representation and easy access to various events.

The MusicalWork class represents individual musical works within the Linked Classical application. It is associated with the Composition class through a one-to-many relationship, signifying that a composition can result in multiple musical works. Conversely, each musical work is linked to one specific composition. Moreover, the MusicalWork class exhibits a recursive one-to-many relationship with itself. This indicates that a musical work can have multiple parts, and each part is itself represented as a MusicalWork object, allowing for a hierarchical representation of complex musical compositions. Furthermore, there is a one-to-many relationship between MusicalWork and Document. This relationship enables multiple documents to be associated with a musical work, facilitating the retrieval of various resources related to a composition.

The Pianist and Conductor classes are present in the application, but there are no relationships for them. They may serve as relevant entities in the system, and further development might incorporate their connections with other classes.

5.5.2 Logical Data Model

Figure 5.3 presents the logical data model inferred from the DBTune classical dataset, representing the actual implementation of the data model in the Linked Classical application. This implementation adheres to the principles of the semantic web, representing data in the form of triples. Each triple consists of a subject, predicate, and object, represented by the resources presented in elliptical shapes, literals represented in rectangles, and arrows signifying predicates. The logical data model effectively maps the entities, attributes, and relationships from the class diagram (Figure 5.2) to triples, enabling seamless integration and navigation within the knowledge graph. URIs uniquely identify resources, ensuring consistency and interlinking with other datasets in the linked data ecosystem.

5.6 Labels Architecture

The Linked Classical application leverages the importance of the `rdfs:label` property in identifying resources with human-readable names on the semantic web. `rdfs:label` is a fundamental property in RDF that allows developers to assign user-friendly labels to resources, making them easily identifiable and understandable by users [16]. By utilizing `rdfs:label`, the Linked

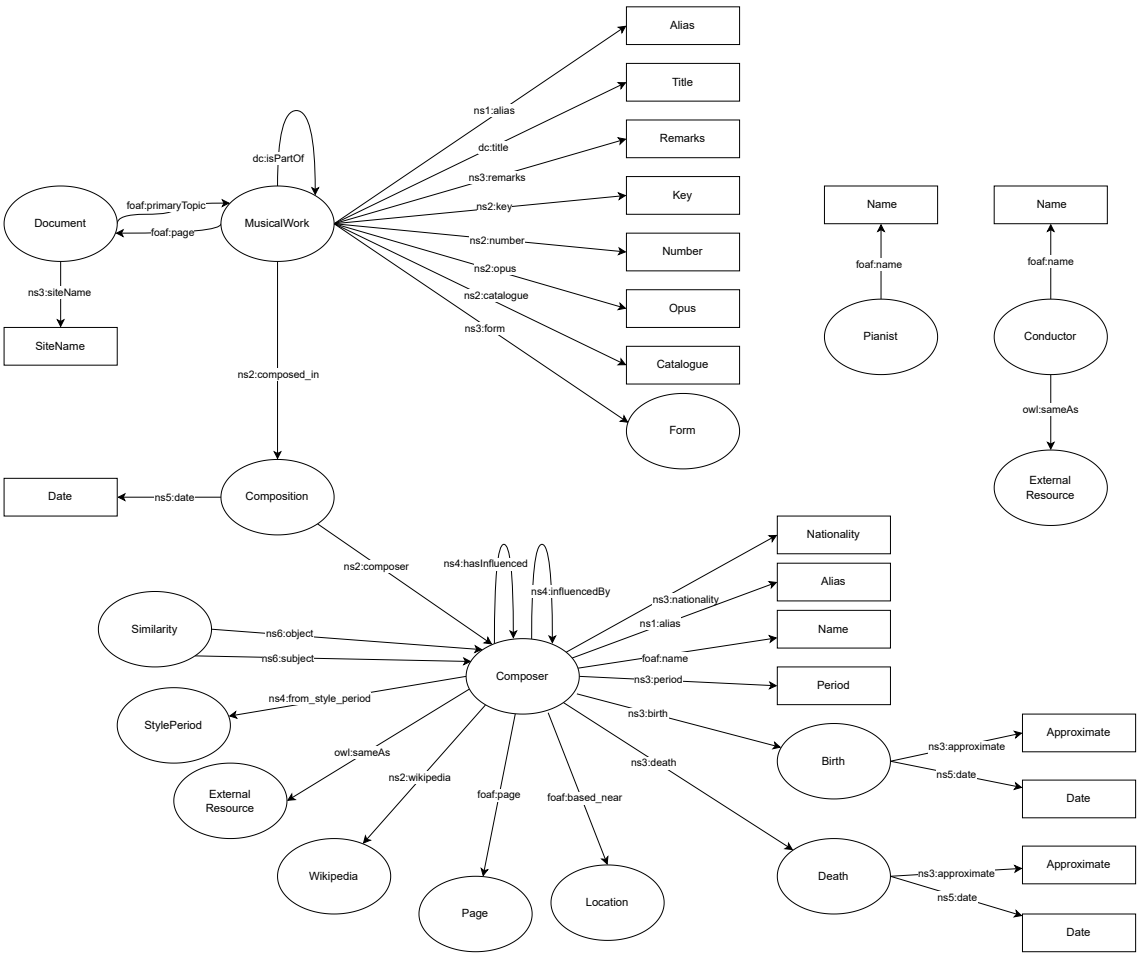

Figure 5.3: Linked Classical Logical Model.

Classical application enhances the user experience and simplifies resource referencing within the knowledge graph.

In the context of the Linked Classical application, we assigned labels to specific types of resources to improve their readability and accessibility. The labels were introduced to the main types of resources of the application, namely to composers, musical works, births and deaths.

To derive the `rdfs:label` for each resource, new triples were created, adhering to the following structure: `<resource-uri> rdfs:label <label>`. The label field was defined according to the type of the resource. For composers, the label was derived from their names, utilizing the `foaf:name` attribute. This approach ensures that composers are easily recognizable by their commonly known names, facilitating navigation and exploration within the application. For musical works, the label was derived from their titles, using the `dc:title` attribute. By providing a human-readable version of a musical work's title, users can effortlessly identify and access specific compositions of interest. For births and deaths, the label was derived from their dates, utilizing the `ns5:date` attribute. This allows users to quickly understand significant events related to composers' lives, providing historical context and chronological information.

The introduction of labels significantly impacted the user interface of the Linked Classical application. Instead of referring to resources using their complex URIs, which are not easily readable, the application displays the assigned labels. This change greatly improves user interaction with the application, as users can easily identify and select composers, musical works, and events of interest.

5.7 URIs Architecture

In the context of the Linked Classical application, the task of managing resource identifiers, known as URIs, becomes pivotal. Although the existing knowledge base resources already possessed URIs, as established by DBTune, the application demanded a more tailored approach to better accommodate its unique requirements and functionalities. The integration of novel URIs into the dataset was executed through the creation of new triples, according to the following structure:

```
<new-URI> owl:sameAs <old-URI>
```

Here, the utilization of the `owl:sameAs` property serves to denote the equivalence of the subject and object resources. This strategy effectively aligned the newly introduced URIs with their corresponding resources. The design of these fresh URIs adhered to a standardized format:

```
http://example.org/linkedclassical/resource/<uuid>
```

In this structure, the `uuid` component represents a universally unique identifier, systematically generated for each new URI, ensuring the distinctiveness and integrity of each identifier. To streamline the utilization of these generated URIs, a direct association was established between

the generated UUIDs and the respective resources, optimizing the accessibility and retrievability of resources. This association was realized through the creation of triples using the `rdf:id` property, following the structure:

```
<resource-uri> rdf:id <uuid>
```

In the application's frontend, these universally unique identifiers played a central role in forming coherent routes. The generated UUIDs provided the basis for the routes of the resource, composer and musical work pages. The routes on the frontend application adhere to the following format, where `<type>` is the type of the resource, which can be `resource`, `composer` or `musicalWork`:

```
https://example.org/linkedclassicalclient/<type>/<uuid>
```

5.8 SPARQL Federation

This section elucidates the practical implementation of SPARQL federation within the Linked Classical application. It delves into the technical intricacies of leveraging SPARQL federation to enhance user experiences by integrating geographical insights and seamlessly merging information from external datasets.

5.8.1 Composer's Geographical Insights

To take advantage of the existence of URIs pointing to external knowledge bases on our dataset, we can use federation techniques to retrieve additional information about a resource. To showcase this capability, we considered the triples present in our knowledge base that have the following structure:

```
<composer-uri> foaf:based_near <location-geonames-uri>
```

These triples employ the `foaf:based_near` predicate [72], serving as a linkage between a composer and a geographical location. The subject is the composer URI and the object is the URI of the location in GeoNames, a user-editable database containing geographical locations [73]. This geographical database is used for retrieving information about the locations associated with a composer, namely its name, coordinates and population. With that retrieved information, we then build a map using Leaflet, an open-source JavaScript library for building interactive maps [74].

As an example, let's consider the following triple about the composer Beethoven:

```
composer:beethoven foaf:based_near location:2921044
```

When the user accesses the page of the composer Beethoven, in the frontend of the Linked Classical application, the SPARQL query of Listing 5.1 is executed in the backend, retrieving information about the coordinates, name, and population of its associated location. This is a

federated SPARQL query, as it uses the `SERVICE` keyword, from SPARQL 1.1, to query a remote SPARQL endpoint, in this case, `http://linkedgeodata.org/sparql`, which is a linked data source for geographical information. This federated approach allows the query to gather data from multiple sources to construct a comprehensive result.

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX gn: <http://www.geonames.org/ontology#>
PREFIX wgs84_pos: <http://www.w3.org/2003/01/geo/wgs84_pos#>

SELECT ?location ?latitude ?longitude ?locationName ?
   locationPopulation
WHERE {
  ?composer rdf:id "0bde8c6b-6695-47b8-933b-01aa4c53cc74" .
  ?composer foaf:based_near ?location .
  SERVICE <http://linkedgeodata.org/sparql> {
    ?location wgs84_pos:lat ?latitude ;
    wgs84_pos:long ?longitude ;
    gn:name ?locationName .
    OPTIONAL { ?location gn:population ?locationPopulation . }
  }
}
```

Listing 5.1: SPARQL query to retrieve additional information about a location.

The outcome materializes in Figure 5.4, depicting a dynamic map that visually conveys this synthesized information. User interaction, such as selecting a location, unveils pertinent details such as the location's name, population, and GeoNames URI. Users can further explore these locations by clicking the GeoNames URI link, which subsequently redirects to a comprehensive repository of location-based information.

5.8.2 Integration with External Datasets

Within the Linked Classical application, a collaborative section within composer pages invites user engagement through the association of external resource that are the same as the respective composer resource. This collaboration results in the seamless integration of fresh insights, thereby enriching the interface's informational landscape. Figure 5.5 provides a visual representation of this interface's functionality. The user chooses the external resource URI and its datasource and the application integrates new information retrieved from that datasource.

To store the necessary metadata to use SPARQL federation, such as the SPARQL endpoints of the external datasets, we used the void vocabulary [36]. This vocabulary describes an RDF dataset using the Resource Description Framework (RDF) data model. It defines a set of terms

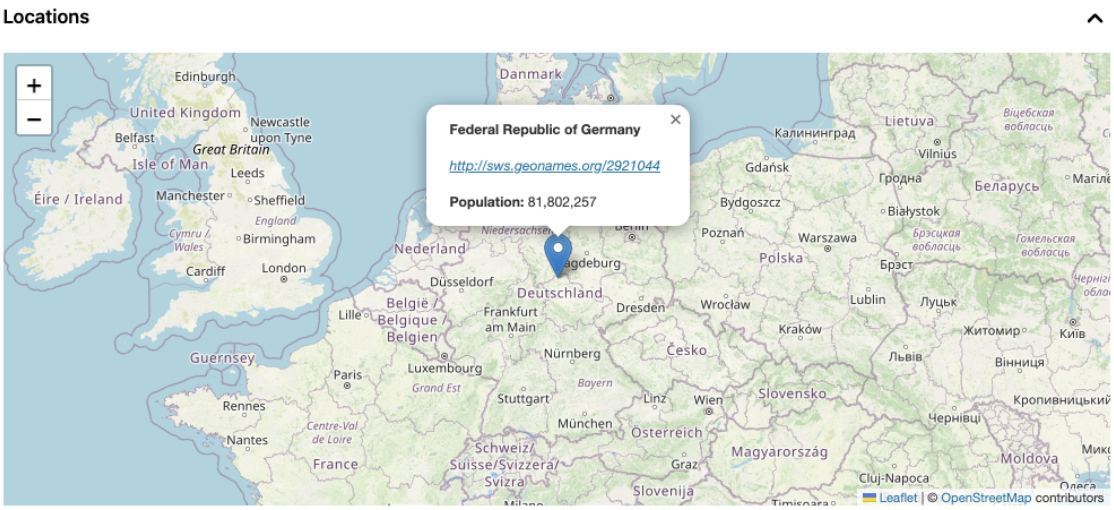

Figure 5.4: Location map for the composer Beethoven.

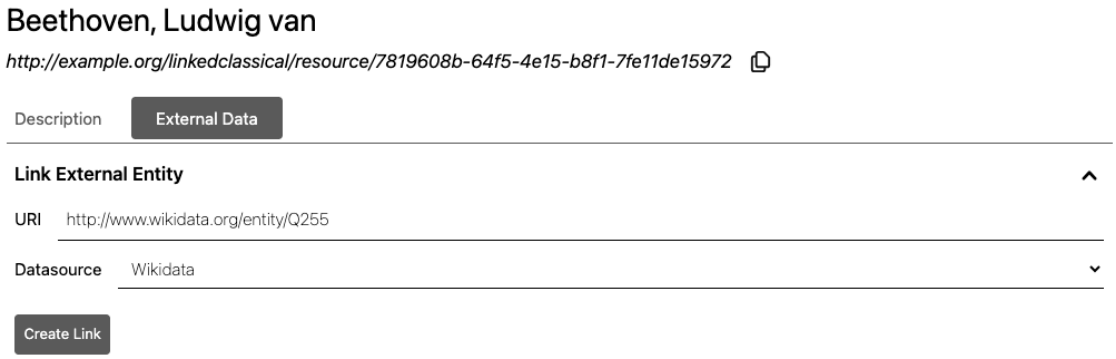

Figure 5.5: Link external entity section.

and properties that can provide metadata about an RDF dataset, such as its name, the entities or concepts described in the dataset, and the URLs of other related datasets. This metadata is stored in an RDF file called void.ttl available in Listing 5.2, which, in addition to storing metadata about the external datasets, can be used to help other users or applications discover and understand our dataset.

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix void: <http://rdfs.org/ns/void#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix LinkedClassical: <https://example.com/linked_classical/> .
@prefix DBpedia: <https://www.dbpedia.org/> .
@prefix Wikidata: <https://www.wikidata.org/> .

## --- our dataset ---

LinkedClassical:  rdf:type void:Dataset ;
        dcterms:title "Linked Classical" ;
        void:sparqlEndpoint <https://example.com/linked_classical/
           sparql> ;
        void:exampleResource <https://example.com/linked_classical/
           composer/97e88489-3475-4e60-b823-1f81fb3132a4> .

## --- datasets to link to ---

# interlinking to DBpedia:
DBpedia:    rdf:type void:Dataset ;
            foaf:homepage <https://www.dbpedia.org/> ;
            dcterms:title "DBPedia" ;
            void:exampleResource <http://dbpedia.org/resource/
               Ludwig_van_Beethoven> ;
            void:sparqlEndpoint <https://dbpedia.org/sparql/> .

LinkedClassical:linked_classical-dbpedia-sameAs  rdf:type void:
   Linkset ;
                           void:linkPredicate owl:sameAs ;
                           void:target LinkedClassical: ;
                           void:target DBpedia: .

# interlinking to Wikidata:
```

```
Wikidata:   rdf:type void:Dataset ;
            foaf:homepage <https://www.wikidata.org/> ;
            dcterms:title "Wikidata" ;
            void:exampleResource <https://www.wikidata.org/entity/
               Q12368917> ;
            void:sparqlEndpoint <https://query.wikidata.org/sparql>
               .

LinkedClassical:linked_classical-sameAs rdf:type void:Linkset ;
                           void:linkPredicate owl:sameAs ;
                           void:target LinkedClassical: ;
                           void:target Wikidata: .
```

Listing 5.2: void.ttl file.

We used the `owl:sameAs` property to make the connection between the resource in the Linked Classical internal dataset and the external entity, stating that these resources are the same [17]. These triples have the following structure:

```
<linked-classical-uri> owl:sameAs <external-entuity-uri>
```

As an example, let's consider again the composer Beethoven, and link its Wikidata resource, which has the URI `http://www.wikidata.org/entity/Q255`. After entering the respective URI and choosing the correct datasource, in this case, Wikidata, the Linked Classical application unveils previously inaccessible insights from Wikidata. A graphical representation of this integration is depicted in Figure 5.6.

The SPARQL query that was executed to retrieve the information from the External Entities section is presented in Listing 5.3. We execute this federated query to present a summary of the main facts about the entity, such as its English label, type, and description.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX void: <http://rdfs.org/ns/void#>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX schema: <http://schema.org/>

SELECT DISTINCT ?labelEn ?instanceOf ?instanceOfLabel ?description
WHERE {
  <http://example.org/linkedclassical/resource/6b7a9889-df36-40b7-
     af42-5ca8ef8a4854> owl:sameAs ?externalEntity .
  ?externalEntity void:inDataset ?dataset .
  ?dataset void:sparqlEndpoint ?sparqlEndpoint .
  SERVICE ?sparqlEndpoint {
```

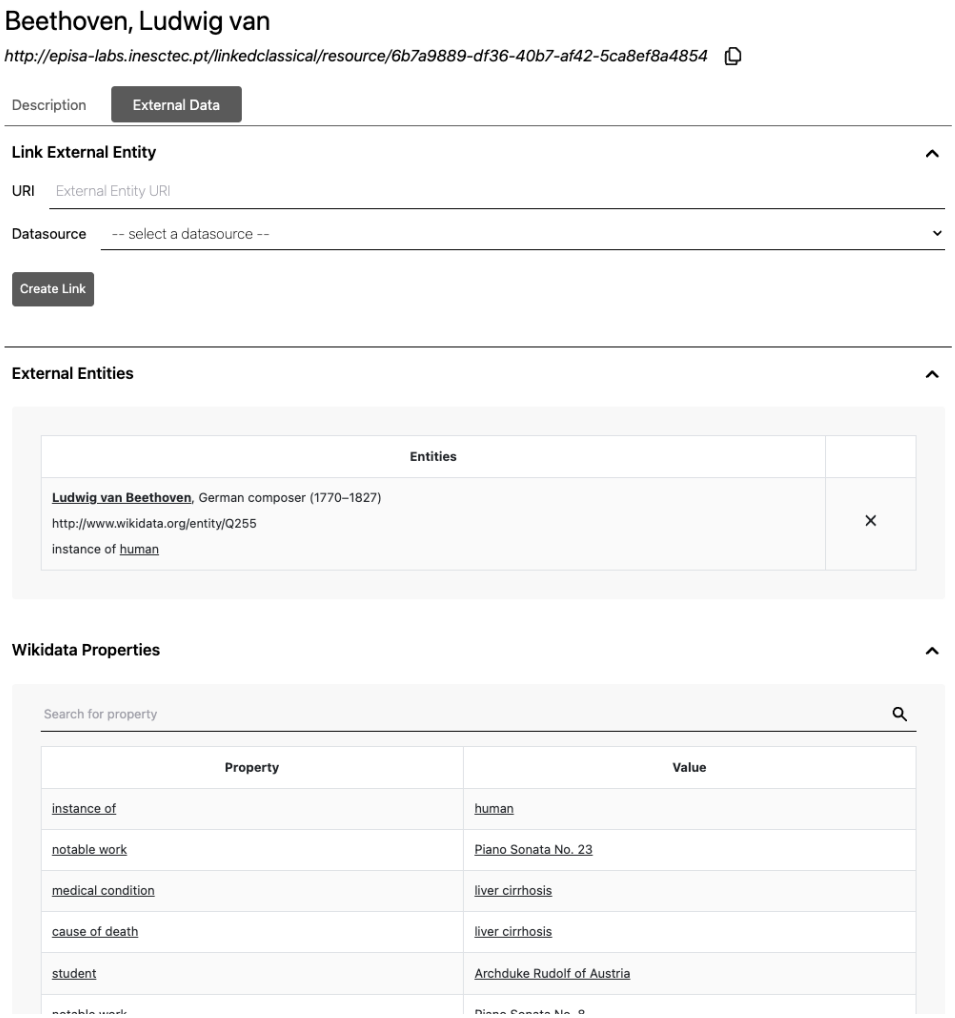

Figure 5.6: External entities and Wikidata properties sections for the composer Beethoven.

```
    ?externalEntity rdfs:label ?labelEn .
    filter (lang(?labelEn) = "en") .
    ?externalEntity wdt:P31 ?instanceOf .
    ?instanceOf rdfs:label ?instanceOfLabel .
    filter (lang(?instanceOfLabel) = "en") .
    ?externalEntity schema:description ?description .
    filter (lang(?description) = "en") .
  }
}
```

Listing 5.3: SPARQL query to retrieve summary information about the composer Beethoven from Wikidata.

The Wikidata Properties section, visible in Figure 5.6, presents all the triples from Wikidata where the subject is the composer. In this section, the user can search for properties, making it easier for finding relevant information. The SPARQL query executed to retrieve the information from this section is presented in Listing 5.4. The query retrieves all the composer information available on Wikidata, namely the predicate and objects or values of all its linked data, as well as its English labels, when available. Note that the `?dataset void:sparqlEndpoint ?sparqlEndpoint` triple pattern is used to get the Wikidata SPARQL endpoint from the metadata fed into our knowledge graph using the `void.ttl` file. Then, the `?sparqlEndpoint` variable is used after the `SERVICE` keyword to reach the Wikidata knowledge base. The `OPTIONAL` keyword is used to ensure the query does not fail when the predicate or object English label does not exist.

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX void: <http://rdfs.org/ns/void#>
PREFIX wikibase: <http://wikiba.se/ontology#>

SELECT DISTINCT ?property ?propertyLabel ?value ?valueLabel
WHERE {
  <http://example.org/linkedclassical/resource/7819608b-64f5-4e15-
     b8f1-7fe11de15972> owl:sameAs ?externalEntity .
  ?entity owl:sameAs ?externalEntity .
  ?externalEntity void:inDataset ?dataset .
  ?dataset void:sparqlEndpoint ?sparqlEndpoint .
  SERVICE ?sparqlEndpoint {
    ?externalEntity ?property ?value .
    ?propertyEntity wikibase:directClaim ?property .
    OPTIONAL {
      ?propertyEntity rdfs:label ?propertyLabel .
```

```
      FILTER (lang(?propertyLabel) = "en") .
    }
    OPTIONAL {
      ?value rdfs:label ?valueLabel .
      FILTER (lang(?valueLabel) = "en") .
    }
    FILTER ( IF (isLiteral(?value), lang(?value) = "en" || lang(?
       value) = "", TRUE) ) .
  }
}
```

Listing 5.4: SPARQL query to retrieve additional information about the composer Beethoven from Wikidata.

We only integrated the Linked Classical application with Wikidata, to show the potential benefits of the integration with external datasources using federation. However, we can integrate other datasources such as DBpedia in the future by adapting the federated SPARQL queries to the respective datasource vocabulary.

Chapter 6

Linked Classical Evaluation

This chapter presents the evaluation of the Linked Classical application concerning the defined user stories outlined in Table 5.1. The evaluation aims to validate the application's alignment with the specified requirements and the extent to which it fulfills the expectations set by the user stories.

6.1 Methodology for Evaluation

The evaluation process focused on assessing the Linked Classical application's functionality in line with the predefined user stories. The approach involved systematically defining and executing test scenarios to validate the expected behavior of the application. The methodology encompassed the following steps:

1. Test Scenarios Definition: Each user story was translated into specific test scenarios that captured the intended interactions and outcomes.

2. Scenario Execution: The defined test scenarios were systematically executed within the application. The interactions were simulated according to the user stories, and the resulting behavior was observed.

3. Outcome Verification: The actual outcomes of each test scenario were compared against the expected outcomes specified in the user stories. This step aimed to verify the correctness and alignment of the application's behavior.

6.2 User Story Evaluation

In this section, the evaluation of the user stories listed in Table 5.1 is presented. The alignment between the application's behavior and the predefined user stories is discussed.

U01: As a user, I want to see the page of a generic resource so that I can see its outgoing and incoming relations

Description

The purpose of this user story is to verify that users can access the page of a generic resource, which is not a composer or a musical work, and view its outgoing and incoming relations.

Test Scenario Definition

Scenario Description: Users attempt to access a generic resource's page and view its outgoing and incoming relations.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. On the birth section, click on the birth URI.

Expected outcome: The application should display the resource's page, providing information about the resource's outgoing and incoming relations.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.1 presents a screenshot of the resource page.

Outcome Verification

Expected outcome: The application should display the resource's page with its outgoing and incoming relations.

Actual outcome: Upon clicking the resource URI, the application successfully redirected the user to the resource's page, displaying the expected information about its relations.

Discussion

The successful execution of this scenario validates that users can seamlessly navigate to the page of a dataset resource by clicking on the resource URI. The displayed information about the resource's outgoing and incoming relations enhances users' understanding of its context within the knowledge graph.

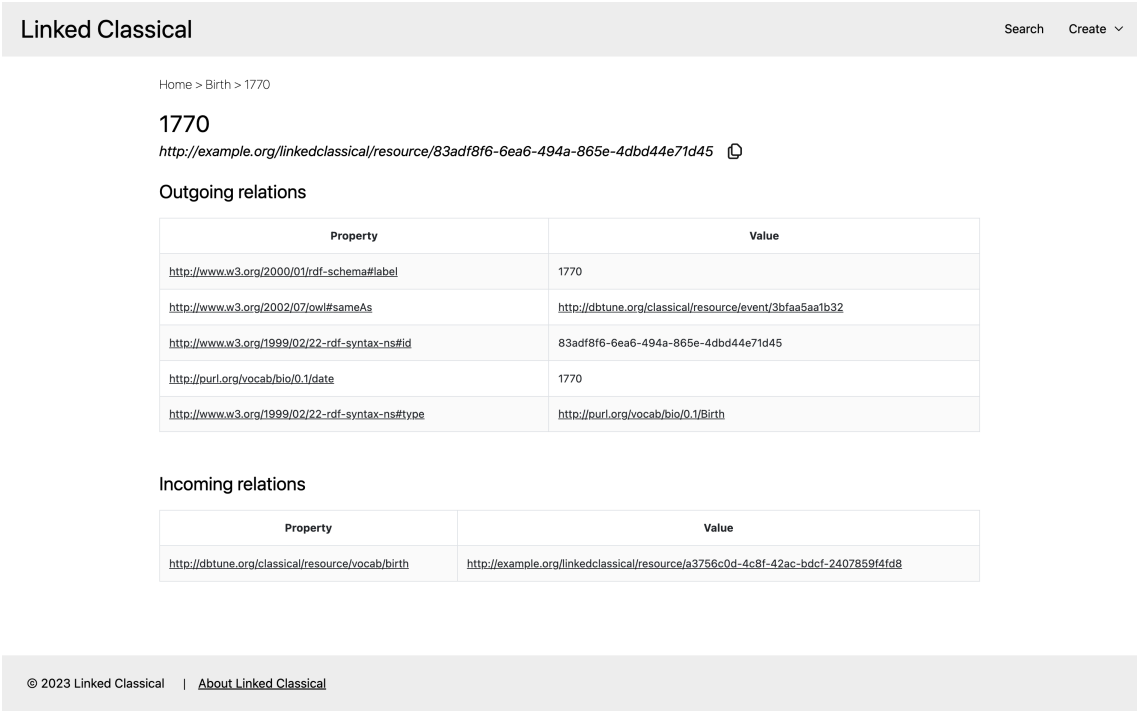

Figure 6.1: Generic resource page.

U02: As a user, I want to see a custom page when the resource I'm viewing is a composer or a musical work so that I can see its information in a more user-friendly way

Description

This user story aims to validate the implementation of custom pages for composers and musical works, offering a user-friendly presentation of information.

Test Scenario Definition 1: Composer Custom Page

Scenario Description: Users access the page of a composer and verify if the information is presented in a more user-friendly manner.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".

Expected outcome: The application should display a custom page for composers, presenting their information in a user-friendly format. The interface should include sections presenting the composer's nationalities, birth, death, web pages, locations, style periods, the composers that he has influenced, and the composers that were influenced by him.

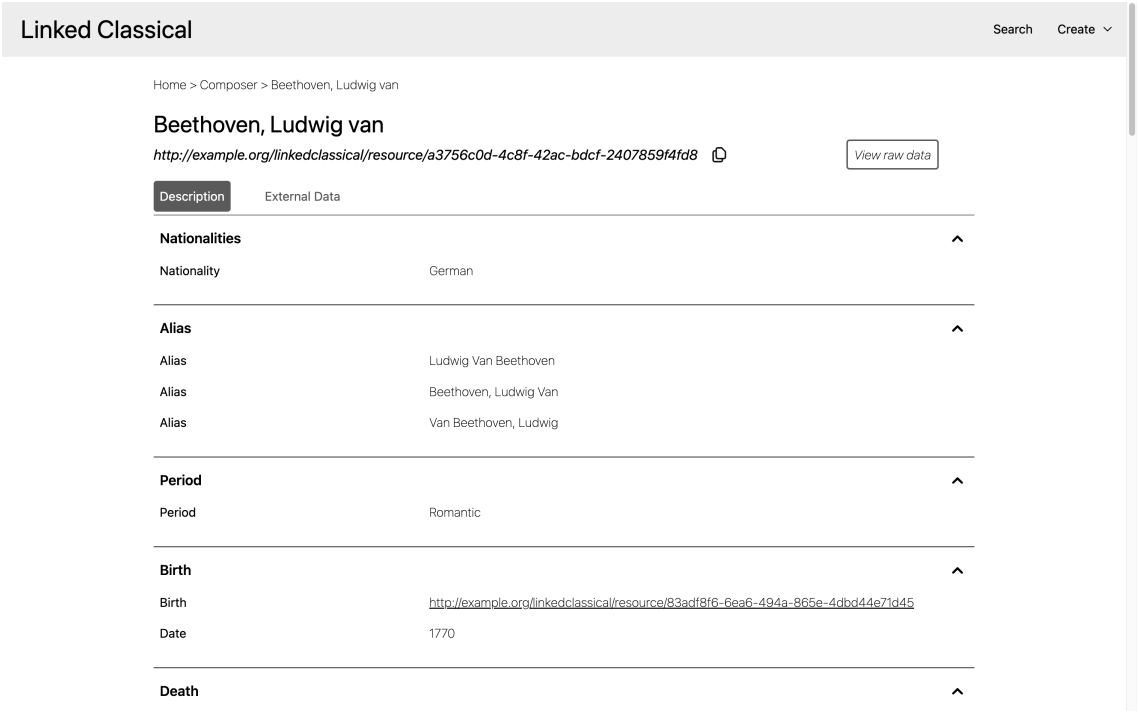

Figure 6.2: Composer Page (1/3).

Scenario Execution 1

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figures 6.2, 6.3 and 6.4 present a screenshot of the composer page.

Outcome Verification 1

Expected outcome: The application should display a custom page for composers, presenting their information in a user-friendly format.

Actual outcome: Upon entering the page of a composer, the application presented a customized page with user-friendly formatting of information.

Test Scenario Definition 2: Musical Work Custom Page

Scenario Description: Users access the page of a musical work and verify if the information is presented in a more user-friendly manner.

Steps:

1. Open the Linked Classical application.
2. Search for the term "moonlight" on the search bar.
3. On the results page, click on the "Musical Works" section.

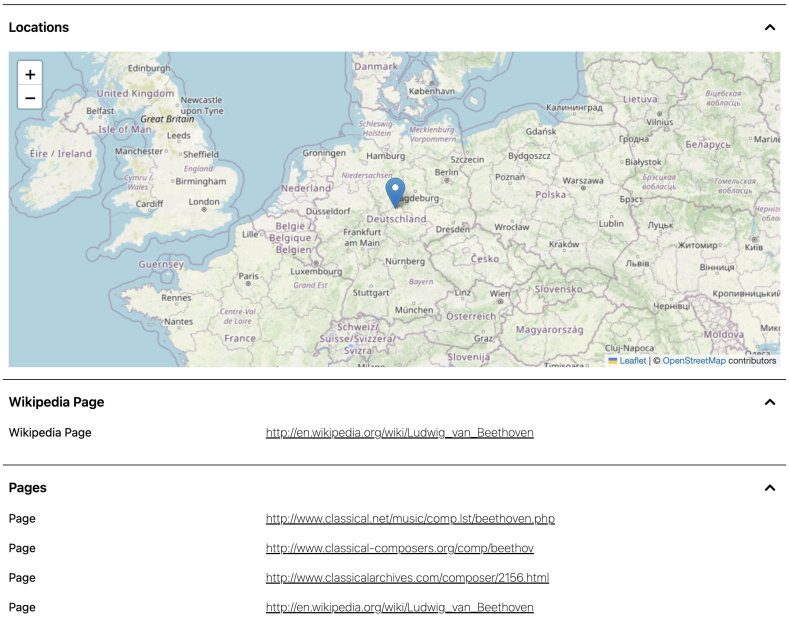

Figure 6.3: Composer Page (2/3).

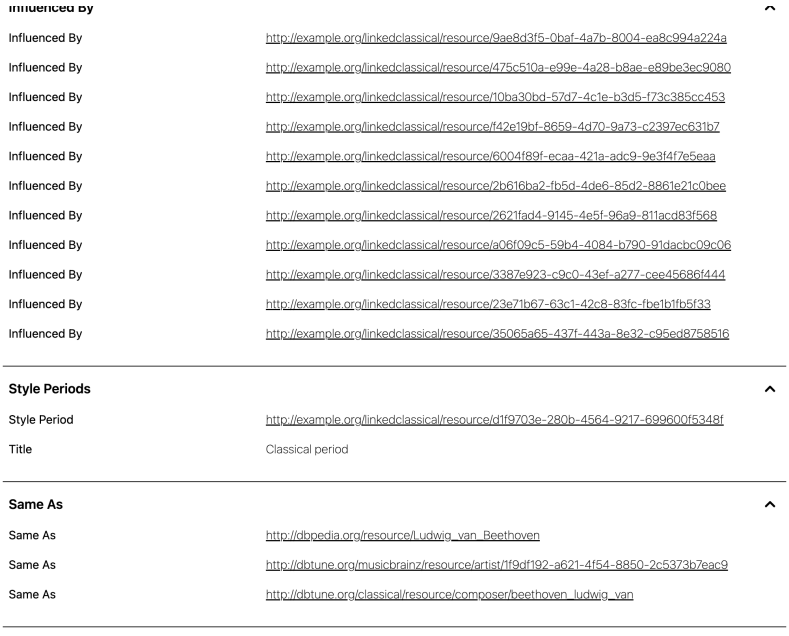

Figure 6.4: Composer Page (3/3).

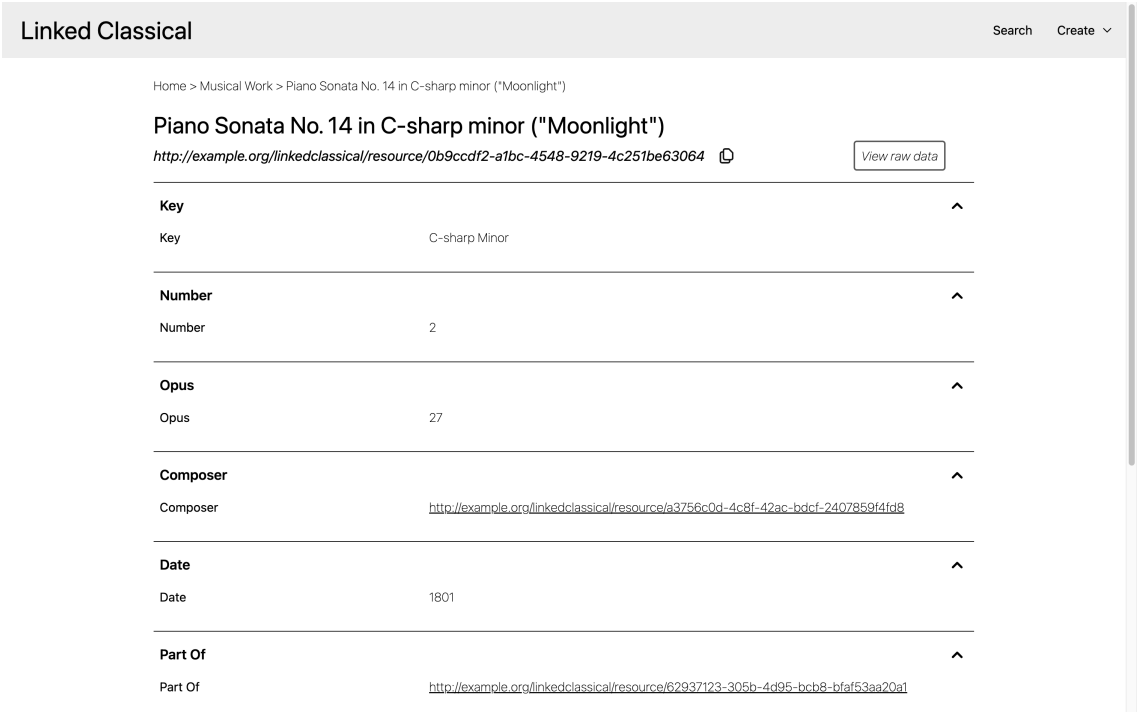

Figure 6.5: Musical work page.

4. Click on the musical work with the title "Piano Sonata No. 14 in C-sharp minor (Moonlight)".

Expected outcome: The application should display a custom page for musical works, presenting their information in a user-friendly format. The interface should include sections presenting the musical work's composer, key, number, opus, date, and related documents.

Scenario Execution 2

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.5 presents a screenshot of the musical work page.

Outcome Verification 2

Expected outcome: The application should display a custom page for musical works, presenting their information in a user-friendly format.

Actual outcome: Upon entering the musical page, the application presented a customized page with user-friendly formatting of information.

Discussion

The successful execution of both scenarios confirms that the Linked Classical application effectively provides custom pages for both composers and musical works, enhancing user experience by offering information in a more user-friendly format.

U03: As a user, I want to click on a resource URI and navigate to its page so that I can see its information

Description

This user story aims to validate the ability of users to click on a resource URI and navigate to the corresponding page to access detailed information.

Test Scenario Definition

Scenario Description: Users click on a resource URI and verify if they are navigated to the corresponding resource page.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. On the birth section, click on the birth URI.
5. Click on the URI value of the incoming relation.

Expected outcome: Clicking on the resource URI should navigate the user to the corresponding resource page, displaying a page according to its type.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior.

Outcome Verification

Expected outcome: Clicking on the resource URI should navigate the user to the corresponding resource page, displaying its detailed information.

Actual outcome: Upon clicking the resource URI, the application successfully navigated the user to the corresponding resource page, displaying relevant information.

Discussion

The successful execution of this scenario verifies that users can effectively navigate to resource pages by clicking on the provided resource URIs, allowing them to access detailed information about the respective resources.

U04: As a user, I want to see a map with the locations where a composer has lived and more information about these locations so that I can learn more about the composer's past

Description

This user story aims to validate the ability of users to view a map displaying the locations where a composer has lived and access additional information about these locations.

Test Scenario Definition

Scenario Description: Users access the composer's page and verify the presence of a map displaying the composer's lived locations, as well as additional information about these locations, such as its name, population, and GeoNames URI.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. Scroll to the "Locations" section.
5. Click on the highlighted location on the map.

Expected outcome: The composer's page should display a map showcasing locations where the composer has lived, along with related information about each location. When the user clicks the location on the map, additional information is shown, such as its name, population, and GeoNames URI.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.6 presents a screenshot of the map after the user has clicked on the location.

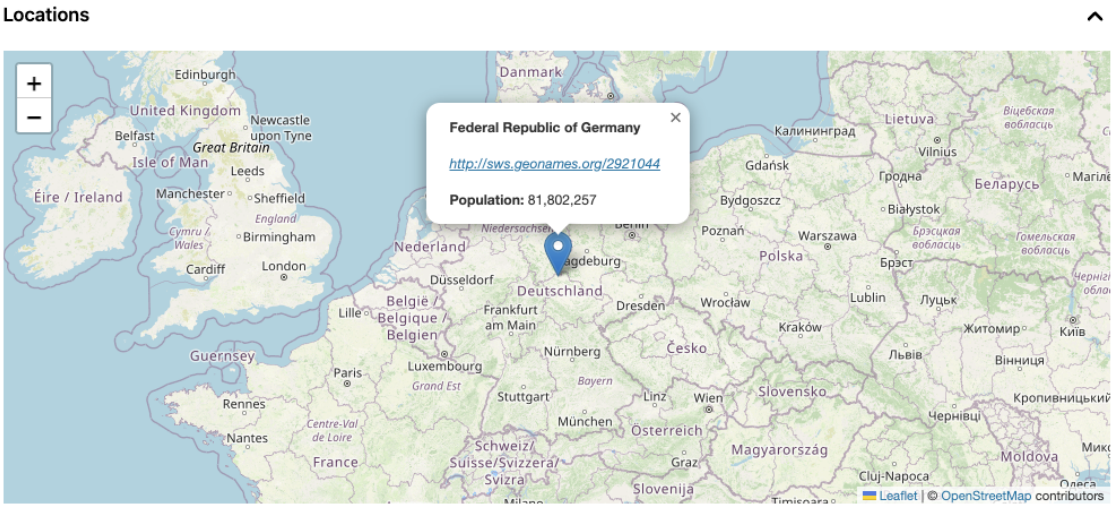

Figure 6.6: Location map.

Outcome Verification

Expected outcome: The composer's page should display a map showcasing locations where the composer has lived, along with related information about each location.

Actual outcome: Upon accessing the composer's page, a map was displayed, indicating the locations where the composer has lived. Clicking on map markers revealed additional information about each location.

Discussion

The successful execution of this scenario confirms that the Linked Classical application provides users with a map that visually represents the locations where a composer has lived, enhancing the user's understanding of the composer's background. The information about the locations was retrieved from external sources, namely GeoNames, using federated SPARQL queries.

U05: As a user, I want to link external entities to the resources of the knowledge graph so that I can enrich the information presented to the users

Description

This user story aims to validate the functionality allowing users to link external entities to resources within the Linked Classical application. This linking process contributes to enriching the information presented to users. This feature was only implemented for composers.

Test Scenario Definition

Scenario Description: Users attempt to link an external entity to an existing composer resource, enhancing the available information about the composer.

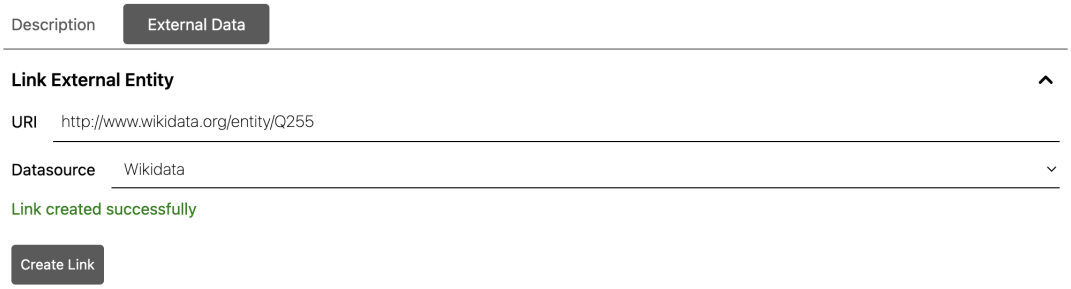

Figure 6.7: Link External Entity section.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. Click on the "External Data" tab.
5. On the "Link External Entity" section, enter the Wikidata URI of the composer Beethoven on the URI field: "http://www.wikidata.org/entity/Q255".
6. Select the datasource as "Wikidata".
7. Click on "Create Link".

Expected outcome: The application should successfully link the provided external entity to the composer resource, enhancing the information associated with the composer.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.7 presents a screenshot of the Link External Entity section after linking the external entity.

Outcome Verification

Expected outcome: The application should successfully link the provided external entity to the composer resource, enhancing the information associated with the composer.

Actual outcome: Upon providing the URI of an external entity and initiating the linking process, the external entity was successfully linked to the composer resource, enriching the available information.

Discussion

The successful execution of this scenario validates that the Linked Classical application enables users to enhance the information presented about a composer by linking external entities. This feature contributes to the contextual enrichment of composer resources.

U06: As a user, I want to see a section on the resource page with information extracted from external sources so that I can have more context about the information

Description

This user story aims to validate the feature that displays information extracted from external sources on resource pages, namely from Wikidata. This feature enhances the user's understanding of the presented information by providing additional context. As with U05, this feature was only implemented for composers.

Test Scenario Definition

Scenario Description: Users navigate to the page of a composer and verify the presence of information extracted from external sources, namely Wikidata, contributing to a deeper understanding of the composer's background.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. Click on the "External Data" tab.
5. On the "Link External Entity" section, enter the Wikidata URI of the composer Beethoven on the URI field: "http://www.wikidata.org/entity/Q255".
6. Select the datasource as "Wikidata".
7. Click on "Create Link".
8. Scroll to the "Wikidata Properties" section.

Expected outcome: The "Wikidata Properties" section should display information extracted from Wikidata, and the user has the possibility to search for a specific property, such as "Spotify artist ID".

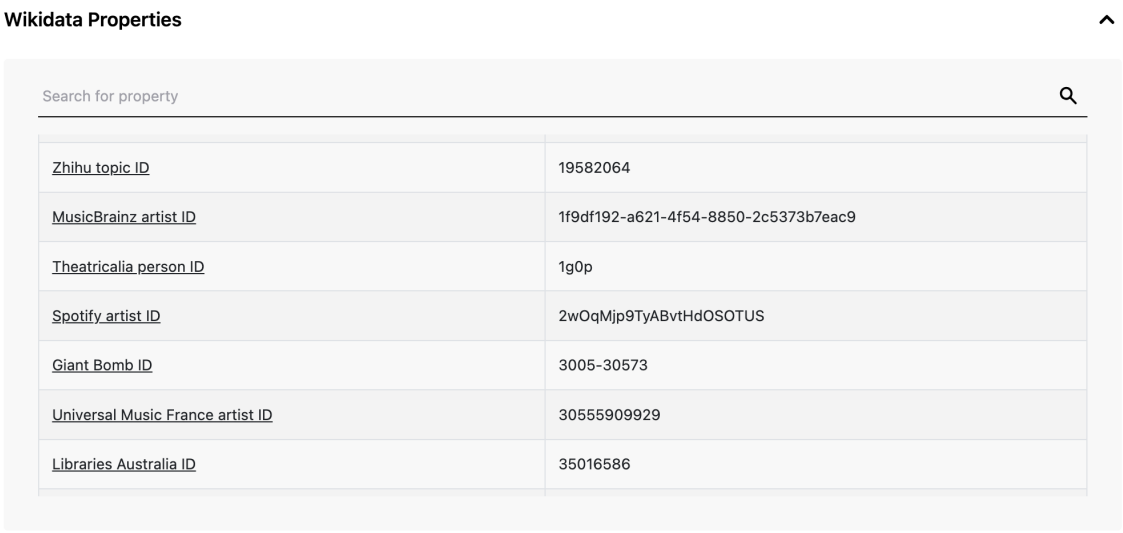

Figure 6.8: Wikidata Properties section.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.8 presents a screenshot of the "Wikidata Properties" section.

Outcome Verification

Expected outcome: After having linked an external entity to the composer resource, the composer page should display information extracted from external sources, providing users with additional context about the composer's background. Furthermore, the user has the possibility to search for specific properties.

Actual outcome: Upon navigating to the composer page after having linked an external entity to the composer resource, the section containing information from external sources was successfully displayed, enhancing the user's understanding of the composer's background. Moreover, the user can search for specific properties in that section.

Discussion

The successful execution of this scenario serves as confirmation that the Linked Classical application can integrate external source information onto resource pages, namely for composers. This augmentation significantly contributes to a more comprehensive and contextually enriched presentation of composer-related information. This implementation empowers users with a broader understanding of a composer's historical context, influences, and overall significance within the realm of music.

It is important to note that although the current scope of this feature centers on composers and is exclusively integrated with Wikidata, its implications are noteworthy across a wider spectrum. This accomplishment sets a promising precedent for future integrations of external data sources, thus indicating the application's potential to provide enriched content from diverse authoritative platforms. By infusing contextual details from Wikidata, the application underscores its adaptability in harnessing reputable data sources to enhance its internal knowledge graph.

U07: As a user, I want to see a section on the resource page with equivalent external entities so that I can look up those entities in other sources

Description

This user story aims to validate that the application presents a section showing what external entities were linked by the users, as well as a summary of the information about each one. As with U05 and U06, this feature was only implemented for composers.

Test Scenario Definition

Scenario Description: Users navigate to the page of a composer and verify the presence of a section presenting the external entities that were added by the users, as well as a summary of its information.

Steps:

1. Open the Linked Classical application.
2. Search for the term "beethoven" on the search bar.
3. On the results page, click on the composer "Beethoven, Ludwig van".
4. Click on the "External Data" tab.
5. On the "Link External Entity" section, enter the Wikidata URI of the composer Beethoven on the URI field: "http://www.wikidata.org/entity/Q255".
6. Select the datasource as "Wikidata".
7. Click on "Create Link".
8. Scroll to the "External Entities" section.

Expected outcome: The "External Entities" section should display the external entities linked by the users, as well as additional information such as their name, description, URI, and type.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.9 presents a screenshot of the "External Entities" section.

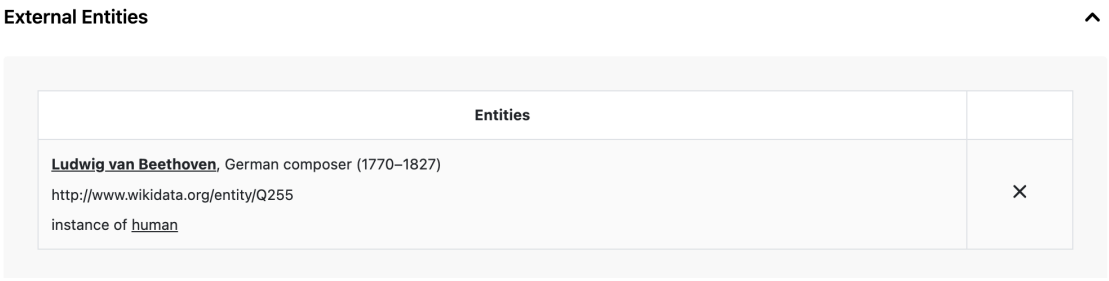

Figure 6.9: External Entities section.

Outcome Verification

Expected outcome: After having linked an external entity to the composer resource, the composer page should display a section presenting the external entities linked by the users, as well as additional information such as their name, description, URI, and type.

Actual outcome: Upon navigating to the composer page after having linked an external entity to the composer resource, the section presenting the external entities linked by the users was successfully displayed. Moreover, additional information was also displayed, namely their name, description, URI, and type.

Discussion

The successful execution of this scenario confirms that the Linked Classical application effectively presents a section with equivalent external entities on the pages of the composers. This feature empowers users to cross-reference and explore linked entities across different sources. While the current implementation focuses on composers, it demonstrates the potential for offering users a broader perspective through connections with external data. Although limited to a specific resource type and data source (Wikidata), the provision of equivalent external entities in the resource section signifies a meaningful step towards expanding the application's contextual richness and aiding users in their research efforts.

U08: As a user, I want to search for composers and musical works so that I can find relevant information about classical music

Description

This user story aims to validate the search functionality within the Linked Classical application, allowing users to search for composers and musical works. The successful execution of this scenario ensures that users can effectively retrieve relevant information about classical music through the application's search feature.

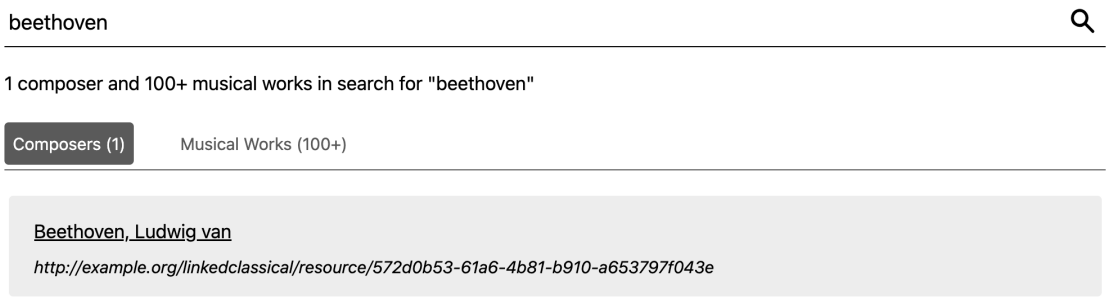

Figure 6.10: Search Results page - Composers tab.

Test Scenario Definition

Scenario Description: Users search for composers and musical works using the application's search functionality and verify the accuracy and relevance of the search results.

Steps:

1. Open the Linked Classical application.
2. Enter the term "beethoven" in the search bar.
3. Press the "Search" button.

Expected outcome: The search results page should display a list of composers and musical works related to the term "beethoven". The "Composers" tab should display the composer "Beethoven", while "Musical Works" tab should present musical works composed by Beethoven.

Scenario Execution

The defined steps were executed within the Linked Classical application to validate the outcome against the expected behavior. Figure 6.10 presents a screenshot of the "Composers" tab on the search results page, while Figure 6.11 presents a screenshot of the respective "Musical Works" tab.

Outcome Verification

Expected outcome: Upon searching for the term "beethoven," the search results page should present the composer Beethoven and musical works composed by this composer.

Actual outcome: After searching for the term "beethoven," the search results page displayed the composer Beethoven and musical works composed by him. The presented results accurately represented the user's search query.

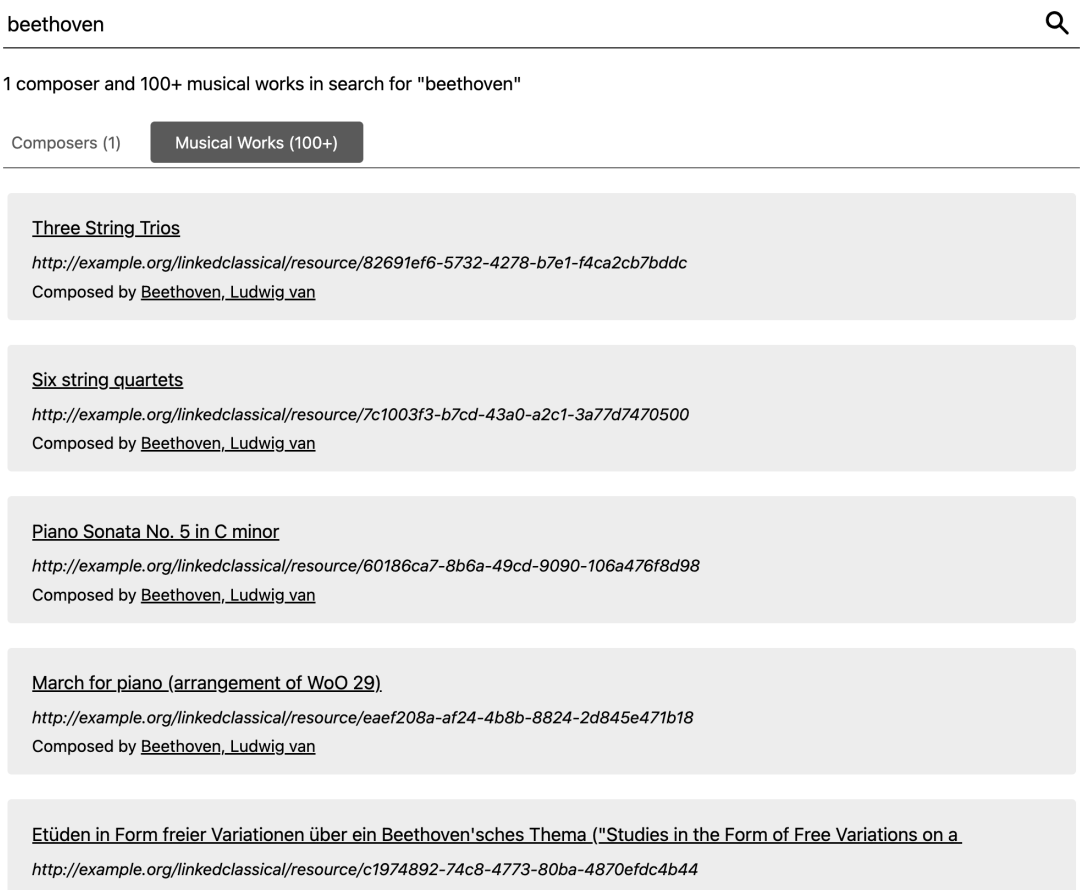

Figure 6.11: Search Results page - Musical Works tab.

Discussion

The successful execution of this scenario demonstrates that the Linked Classical application's search functionality effectively facilitates users' exploration of composers and musical works. Users can confidently retrieve information about classical music entities through the search feature. By offering accurate and relevant search results, the application enhances users' ability to discover and access valuable information within the domain of classical music.

6.3 Results and Observations

The evaluation process yielded the following observations and results:

- Resource Page Navigation: The application successfully enabled users to access the page of a dataset resource (U01) and navigate to it by clicking on the resource URI (U03).
- Custom Composer and Musical Work Pages: Users were presented with custom pages for composers and musical works (U02) that enhanced the presentation of relevant information.
- Map-based Insights: The application effectively displayed maps with composer location insights (U04), offering users additional context about the geographical aspects of composers' lives.
- External Entity Linking: The application supported linking external entities to enrich resource information (U05), facilitating the presentation of contextual details.
- Information Extraction from External Sources: Users could access information extracted from external sources (U06) to gain additional context and insights.
- Equivalence Across Sources: The application displayed equivalent external entities (U07), allowing users to cross-reference information from other sources.
- Effective Search Functionality: Users could successfully search for composers and musical works (U08), facilitating information retrieval.

6.4 Discussion

The evaluation revealed that the Linked Classical application aligns well with the predefined user stories. The application's behavior in response to different scenarios was consistent with the expected functionality outlined in the stories.

While the application successfully met the user story requirements, opportunities for future enhancements and refinements were identified. These include refining the presentation of information from external sources, integrating with additional external sources, and implementing federation features for other types of resources beyond composers.

Chapter 7

Conclusion

In this final chapter, we summarize the key findings and contributions of the present study. Moreover, we delve into the implications of the research's outcomes and how they can shape future investigations in the field of linked data applications that make use of federation.

7.1 Conclusions and Main Contributions

The rise of the semantic web marks a significant change in how we can represent and comprehend data. This paradigm shift has driven the development of various innovative applications that take advantage of the power of structured and interlinked data. However, the journey toward fully harnessing the potential of linked data has encountered substantial challenges, particularly in the field of data integration. Our research has focused on these challenges, highlighting the necessity of effective data integration techniques to overcome the limitations imposed by fragmented data sources and divergent vocabularies.

Through the development of the Linked Classical application, we demonstrated the viability of a solution that integrates data from external knowledge bases into an application. This solution, rooted in the utilization of SPARQL query federation, enables the aggregation of information from diverse sources, enriching the user experience by increasing the understanding of the domain. After the creation and evaluation of the Linked Classical application, we can conclude that using federation to integrate data available on the semantic web can substantially enhance a linked data application by giving its users access to more useful and detailed information according to the usage context, validating the hypothesis formulated in Section 4.1.

Another key contribution is the publication of the Linked Classical application as an open-source project. By publishing our solution, we offer a practical example for building similar applications that harness the potential of linked data integration. Collectively, these contributions not only advance our understanding of data integration challenges within linked data applications but also provide concrete tools and insights to foster the growth and adoption of semantic web technologies across diverse domains.

7.2 Further Work

Looking ahead, our research points to several promising directions for further exploration and refinement, aimed at advancing the practical implementation of federation solutions for linked data applications. A significant area for future focus lies in enhancing user interaction and experience. Developing innovative visualization techniques that simplify the integration of information from external sources could substantially improve the accessibility and usability of federation techniques in linked data applications. By creating interfaces that are intuitive and user-friendly, we can abstract the technical complexities of linked data, ensuring that a wider audience can seamlessly navigate and extract value from the interconnected information.

Dealing with scalability and performance issues is another vital aspect that requires ongoing focus. As knowledge graphs grow larger and more intricate, it becomes essential to optimize SPARQL federated queries. Exploring methods to improve query speed and efficiency will be pivotal in addressing challenges related to querying extensive datasets while maintaining the necessary responsiveness for smooth user interactions.

Furthermore, expanding the applicability of our solution to domains beyond classical music shows great potential. By showcasing its applicability across various domains, we can promote broader adoption and uncover novel linked data applications that benefit from federation techniques.

Bibliography

[1] Tim Berners-Lee, Robert Cailliau, Jean-François Groff, and Bernd Pollermann. World-Wide Web: The Information Universe. *Internet Research*, 20:461–471, 08 2010.

[2] Sareh Aghaei, Mohammad Ali Nematbakhsh, and Hadi Khosravi Farsani. Evolution of the world wide web: From Web 1.0 to Web 4.0. *International Journal of Web & Semantic Technology*, 3(1):1–10, 2012.

[3] Tim Berners-Lee, James Hendler, and Ora Lassila. The Semantic Web. *Scientific American*, 284(5):34–43, 2001.

[4] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked Data - The Story So Far. *Int. J. Semantic Web Inf. Syst.*, 5(3):1–22, 2009.

[5] Florian Bauer and Martin Kaltenböck. Linked open data: The essentials. *Edition mono/-monochrom, Vienna*, 710, 2011.

[6] Open Data Handbook. What is Open Data? https://opendatahandbook.org/guide/en/what-is-open-data/. Accessed: 2023-01-03.

[7] Michalis Mountantonakis and Yannis Tzitzikas. Large-scale Semantic Integration of Linked Data: A Survey. *ACM Comput. Surv.*, 52(5):103:1–103:40, 2019.

[8] Muhammad Saleem, Yasar Khan, Ali Hasnain, Ivan Ermilov, and Axel-Cyrille Ngonga Ngomo. A fine-grained evaluation of SPARQL endpoint federation systems. *Semantic Web*, 7(5):493–518, 2016.

[9] EUCLID Project. Chapter 4: Interaction with Linked Data. https://www.euclid-project.eu/modules/chapter4.html. Accessed: 2023-01-04.

[10] Christian Bizer. The Emerging Web of Linked Data. *IEEE Intell. Syst.*, 24(5):87–92, 2009.

[11] Tim Berners-Lee. Linked Data. https://www.w3.org/DesignIssues/LinkedData.html, 2006. Accessed: 2023-01-12.

[12] Insight Centre for Data Analytics. The Linked Open Data Cloud. https://lod-cloud.net/. Accessed: 2023-01-13.

[13] Steve Bratt. Semantic Web, and Other Technologies to Watch, slide 24. https://www.w3.org/2007/Talks/0130-sb-W3CTechSemWeb/#(24). Accessed: 2023-01-14.

[14] Carroll J. Klyne G. Resource Description Framework (RDF): Concepts and Abstract Syntax - W3C Recommendation. https://www.w3.org/TR/rdf-concepts/, 2004. Accessed: 2023-01-14.

[15] Seiji Sakakibara, Sachio Saiki, Masahide Nakamura, and Kiyoshi Yasuda. Generating Personalized Dialogue Towards Daily Counseling System for Home Dementia Care. In Vincent G. Duffy, editor, *Digital Human Modeling. Applications in Health, Safety, Ergonomics, and Risk Management: Health and Safety - 8th International Conference, DHM 2017, Held as Part of HCI International 2017, Vancouver, BC, Canada, July 9-14, 2017, Proceedings, Part II*, volume 10287 of *Lecture Notes in Computer Science*, pages 161–172. Springer, 2017.

[16] R.V. Guha Dan Brickley. RDF Schema 1.1 - W3C Recommendation. https://www.w3.org/TR/rdf-schema/, 2014. Accessed: 2023-01-14.

[17] Frank van Harmelen Deborah L. McGuinness. OWL Web Ontology Language Overview - W3C Recommendation. https://www.w3.org/TR/owl-features/, 2004. Accessed: 2023-01-14.

[18] Andy Seaborne Steve Harris. SPARQL 1.1 Query Language - W3C Recommendation. https://www.w3.org/TR/sparql11-query/, 2013. Accessed: 2023-01-14.

[19] Sergey Brin and Lawrence Page. The Anatomy of a Large-Scale Hypertextual Web Search Engine. *Comput. Networks*, 30(1-7):107–117, 1998.

[20] S. Chakrabarti, B.E. Dom, S.R. Kumar, P. Raghavan, S. Rajagopalan, A. Tomkins, D. Gibson, and J. Kleinberg. Mining the web's link structure. *Computer*, 32(8):60–67, 1999.

[21] Roberto Mirizzi, Azzurra Ragone, Tommaso Di Noia, and Eugenio Di Sciascio. Ranking the Linked Data: The Case of DBpedia. In Boualem Benatallah, Fabio Casati, Gerti Kappel, and Gustavo Rossi, editors, *Web Engineering, 10th International Conference, ICWE 2010, Vienna, Austria, July 5-9, 2010. Proceedings*, volume 6189 of *Lecture Notes in Computer Science*, pages 337–354. Springer, 2010.

[22] David J. Hand. Aspects of Data Ethics in a Changing World: Where Are We Now? *Big Data*, 6(3):176–190, 2018. PMID: 30283727.

[23] Ian Millard, Hugh Glaser, Manuel Salvadores, and Nigel Shadbolt. Consuming Multiple Linked Data Sources: Challenges and Experiences. In Olaf Hartig, Andreas Harth, and Juan F. Sequeda, editors, *Proceedings of the First International Workshop on Consuming Linked Data, Shanghai, China, November 8, 2010*, volume 665 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2010.

[24] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary G. Ives. DBpedia: A Nucleus for a Web of Open Data. In Karl Aberer, Key-Sun Choi, Natasha Fridman Noy, Dean Allemang, Kyung-Il Lee, Lyndon J. B. Nixon, Jennifer Golbeck, Peter Mika, Diana Maynard, Riichiro Mizoguchi, Guus Schreiber, and Philippe Cudré-Mauroux, editors, *The Semantic Web, 6th International Semantic Web Conference, 2nd Asian Semantic Web Conference, ISWC 2007 + ASWC 2007, Busan, Korea, November 11-15, 2007*, volume 4825 of *Lecture Notes in Computer Science*, pages 722–735. Springer, 2007.

[25] Denny Vrandecic and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014.

[26] EUCLID Project. Chapter 5: Building Linked Data Applications. https://www.euclid-project.eu/modules/chapter5.html. Accessed: 2023-01-31.

[27] Michael Martin and Sören Auer. Categorisation of semantic web applications. In *Proceedings of the 4th International Conference on Advances in Semantic Processing (SEMAPRO2010), October*, pages 25–30, 2010.

[28] Dominic Oldman and Diana Tanase. Reshaping the knowledge graph by connecting researchers, data and practices in researchspace. In Denny Vrandečić, Kalina Bontcheva, Mari Carmen Suárez-Figueroa, Valentina Presutti, Irene Celino, Marta Sabou, Lucie-Aimée Kaffee, and Elena Simperl, editors, *The Semantic Web – ISWC 2018*, pages 325–340, Cham, 2018. Springer International Publishing.

[29] David Myers, Alison Dalgity, Ioannis Avramides, and Dennis Wuthrich. Arches: An open source gis for the inventory and management of immovable cultural heritage. In Marinos Ioannides, Dieter Fritsch, Johanna Leissner, Rob Davies, Fabio Remondino, and Rossella Caffo, editors, *Progress in Cultural Heritage Preservation*, pages 817–824, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[30] David Myers, Alison Dalgity, and Ioannis Avramides. The arches heritage inventory and management system: a platform for the heritage field. *Journal of Cultural Heritage Management and Sustainable Development*, 2016.

[31] Doremus – DOing REusable MUSical data. https://www.doremus.org/. Accessed: 2023-01-31.

[32] Overture - Doremus. https://overture.doremus.org/. Accessed: 2023-01-31.

[33] Pasquale Lisena, Manel Achichi, Eva Fernández, Konstantin Todorov, and Raphaël Troncy. Exploring linked classical music catalogs with overture. In *ISWC: International Semantic Web Conference*, 2016.

[34] Tom Heath and Enrico Motta. Revyu: Linking reviews and ratings into the Web of Data. *J. Web Semant.*, 6(4):266–273, 2008.

[35] EUCLID Project. Chapter 3: Providing Linked Data. https://www.euclid-project.eu/modules/chapter3.html. Accessed: 2023-01-28.

[36] Michael Hausenblas Jun Zhao Keith Alexander, Richard Cyganiak. Describing Linked Datasets with the VoID Vocabulary. https://www.w3.org/TR/void/, 2011. Accessed: 2023-01-28.

[37] Open Knowledge Foundation. CKAN - The open source data management system. https://ckan.org/. Accessed: 2023-01-28.

[38] Adam Kariv Rufus Pollock. DataHub - Frictionless Data. https://datahub.io/. Accessed: 2023-01-28.

[39] Olaf Hartig. An overview on execution strategies for Linked Data queries. *Datenbank-Spektrum*, 13:89–99, 2013.

[40] Nur Aini Rakhmawati, Jürgen Umbrich, Marcel Karnstedt, Ali Hasnain, and Michael Hausenblas. Querying over Federated SPARQL Endpoints - A State of the Art Survey. *CoRR*, abs/1306.1723, 2013.

[41] Carlos Buil-Aranda Eric Prud'hommeaux. SPARQL 1.1 Federated Query. https://www.w3.org/TR/sparql11-federated-query/, 2013. Accessed: 2023-01-28.

[42] Peng Peng, Lei Zou, M Tamer Özsu, Lei Chen, and Dongyan Zhao. Processing SPARQL queries over distributed RDF graphs. *The VLDB Journal*, 25:243–268, 2016.

[43] Damla Oguz, Belgin Ergenc, Shaoyi Yin, Oguz Dikenelli, and Abdelkader Hameurlain. Federated Query Processing on Linked Data: A Qualitative Survey and Open Challenges. *The Knowledge Engineering Review*, 30(5):545–563, 2015.

[44] Linked Data Fragments. https://linkeddatafragments.org/. Accessed: 2023-01-31.

[45] Zhenzhen Gu, Francesco Corcoglioniti, Davide Lanti, Alessandro Mosca, Guohui Xiao, Jing Xiong, and Diego Calvanese. A systematic overview of data federation systems. *Semantic Web*, pages 1–59, 12 2022.

[46] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. FedX: Optimization Techniques for Federated Query Processing on Linked Data. In Lora Aroyo, Chris Welty, Harith Alani, Jamie Taylor, Abraham Bernstein, Lalana Kagal, Natasha Fridman Noy, and Eva Blomqvist, editors, *The Semantic Web - ISWC 2011 - 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I*, volume 7031 of *Lecture Notes in Computer Science*, pages 601–616. Springer, 2011.

[47] The Eclipse Foundation. Eclipse RDF4J. https://rdf4j.org/. Accessed: 2023-01-31.

[48] Sibel Adali, K Selçuk Candan, Yannis Papakonstantinou, and Vo S Subrahmanian. Query caching and optimization in distributed mediator systems. *ACM SIGMOD Record*, 25(2):137–146, 1996.

[49] Qingqing Gan and Torsten Suel. Improved techniques for result caching in web search engines. In *Proceedings of the 18th international conference on World wide web*, pages 431–440, 2009.

[50] B Barla Cambazoglu, Ismail Sengor Altingovde, Rifat Ozcan, and Özgür Ulusoy. Cache-based query processing for search engines. *ACM Transactions on the Web (TWEB)*, 6(4):1–24, 2012.

[51] Gregory Todd Williams and Jesse Weaver. Enabling fine-grained HTTP caching of SPARQL query results. In *The Semantic Web–ISWC 2011: 10th International Semantic Web Conference, Bonn, Germany, October 23-27, 2011, Proceedings, Part I 10*, pages 762–777. Springer, 2011.

[52] Michael Martin, Jörg Unbehauen, and Sören Auer. Improving the performance of semantic web applications with SPARQL query caching. In *The Semantic Web: Research and Applications: 7th Extended Semantic Web Conference, ESWC 2010, Heraklion, Crete, Greece, May 30–June 3, 2010, Proceedings, Part II 7*, pages 304–318. Springer, 2010.

[53] Peter Mika, Abraham Bernstein, Chris Welty, Craig Knoblock, Denny Vrandečić, Paul Groth, Natasha Noy, Krzysztof Janowicz, and Carole Goble. *The Semantic Web – ISWC 2014: 13th International Semantic Web Conference, Riva del Garda, Italy, October 19-23, 2014. Proceedings, Part II*, volume 8797. Springer, 2014.

[54] Anna Gossena, Peter Haasea, Christian Hüttera, Michael Meiera, Andriy Nikolova, Christoph Pinkela, Michael Schmidta, Andreas Schwartea, and Johannes Tramea. The Information Workbench – A Platform for Linked Data Applications, 2016.

[55] Data.gov – The home of the U.S. Government's open data. https://data.gov/. Accessed: 2023-01-31.

[56] Guillermo Vega-Gorgojo. Lod4culture: Easy exploration of cultural heritage linked open data. *Semantic Web*, (Preprint):1–30.

[57] Guillermo Vega-Gorgojo. CRAFTS: configurable REST APIs for triple stores. *IEEE Access*, 10:32426–32441, 2022.

[58] INESC TEC. EPISA. https://episa.inesctec.pt/. Accessed: 2023-01-20.

[59] Maria José de Almeida; Inês Koch; Cláudia Guedes. EPISA project: a FAIR path to semantic archives at the Portuguese National Archives. https://www.rd-alliance.org/

episa-project-fair-path-semantic-archives-portuguese-national-archives. Accessed: 2023-01-20.

[60] Cláudia Raquel Amaral Conde Guedes. Designing User Interaction with Linked Data in Historical Archives. Master's thesis, FEUP, October 2020.

[61] International Council on Archives. What are archives? https://www.ica.org/en/what-archive. Accessed: 2023-01-20.

[62] Memorix Archive. ISAD(G): General International Standard Archival Description. https://archives.memorix.nl/standards/ISAD(G). Accessed: 2023-01-20.

[63] Karen Gracy. Archival description and linked data: a preliminary study of opportunities and implementation challenges. *Archival Science*, 15, 02 2014.

[64] Ashleigh Hawkins. Advocating for linked archives: the benefits to users of archival linked data. In *Proceedings of the Linked Archives International Workshop*, pages 52–63, 2021.

[65] GitHub. https://github.com/. Accessed: 2023-06-02.

[66] Laravel - The PHP Framework for Web Artisans. https://laravel.com/. Accessed: 2023-06-02.

[67] Apache Jena Fuseki. https://jena.apache.org/documentation/fuseki2/. Accessed: 2023-06-02.

[68] Spring Boot. https://spring.io/projects/spring-boot. Accessed: 2023-06-02.

[69] Mike Cohn. *User stories applied: For agile software development*. Addison-Wesley Professional, 2004.

[70] DBTune Classical. http://dbtune.org/classical/. Accessed: 2023-05-31.

[71] DBTune. http://dbtune.org/. Accessed: 2023-05-31.

[72] FOAF Vocabulary Specification. http://xmlns.com/foaf/0.1/. Accessed: 2023-05-27.

[73] GeoNames. https://www.geonames.org/. Accessed: 2023-05-27.

[74] Leaflet. https://leafletjs.com/. Accessed: 2023-05-28.