Semantic Web: Past, Present, and Future

Extended Version from 2025-11-11 of TGDK 2(1): 3:1-3:37 (2024)

Ansgar Scherp
Ulm University, Germany

Gerd Groener
Carl Zeiss SMT GmbH, Germany

Petr Škoda
Department of Software Engineering, Faculty of Mathematics and Physics, Charles University, Prague, Czechia

Katja Hose
TU Wien, Austria

Maria-Esther Vidal
Leibniz University of Hannover and TIB-Leibniz Information Centre for Science and Technology, Germany

Abstract

Ever since the vision was formulated, the Semantic Web has inspired many generations of innovations. Semantic technologies have been used to share vast amounts of information on the Web, enhance them with semantics to give them meaning, and enable inference and reasoning on them. Throughout the years, semantic technologies, and in particular knowledge graphs, have been used in search engines, data integration, enterprise settings, and machine learning.

In this paper, we recap the classical concepts and foundations of the Semantic Web as well as modern and recent concepts and applications, building upon these foundations. The classical topics we cover include knowledge representation, creating and validating knowledge on the Web, reasoning and linking, and distributed querying. We enhance this classical view of the so-called "Semantic Web Layer Cake" with an update of recent concepts. These include provenance, security and trust, as well as a discussion of practical impacts from industry-led contributions. We also provide an overiew of shallow and deep machine learning methods for knowledge graphs and discuss the relation of language models and knowledge graphs. We conclude with an outlook on the future directions of the Semantic Web.

2012 ACM Subject Classification Information systems → Semantic web description languages; Information systems → Markup languages; Computing methodologies → Ontology engineering; Computing methodologies → Knowledge representation and reasoning

Keywords and phrases Linked Open Data, Semantic Web Graphs, Knowledge Graphs

Digital Object Identifier 10.4230/TGDK.2.1.2

Category Survey

Related Version Lifelong version
Full Version: `https://github.com/ascherp/semantic-web-primer`

Funding *Maria-Esther Vidal*: Partially funded by Leibniz Association, program "Leibniz Best Minds: Programme for Women Professors", project TrustKG-Transforming Data in Trustable Insights; Grant P99/2020

Received **Accepted** **Published** November 13, 2025

Editor Aidan Hogan, Ian Horrocks, Andreas Hotho, and Lalana Kagal

Special issue Trends in Graph Data and Knowledge – Part 2

Transactions on Graph Data and Knowledge, Vol. 2, Issue 1, Article No. 2, pp. 2:1–2:49
TGDK Transactions on Graph Data and Knowledge
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Changelog

- **Update in November 2025**: Added an overview of *machine learning methods for knowledge graphs* in Section 9, particularly shallow graph embeddings and neural networks for graphs. We discuss *language models* and their relation to and use for knowledge graphs in Section 10.

Preamble

This article is a living document. The first version was written by Jannik, Scherp, and Staab in 2011 [120]. It was translated to German and updated by Gröner, Scherp, and Staab in 2013 [92] and updated again by Gröner and Scherp in 2020 [183]. This release in 2024 reflects on the latest developments in knowledge graphs and large language models. It has been translated back to English and extended with contributions by Petr Škoda, Katja Hose, and Maria-Esther Vidal.

If you like to contribute, please contact the first author and visit: `https://github.com/ascherp/semantic-web-primer`

Please cite this paper as, see `https://dblp.org/rec/journals/tgdk/ScherpGOHV24.html?view=bibtex`:

```
@article{DBLP:journals/tgdk/ScherpGOHV24,
 author = {Ansgar Scherp and Gerd Gr{\"{o}}ner and
  Petr Skoda and Katja Hose and
  Maria{-}Esther Vidal},
 title        = {Semantic Web: Past, Present, and Future},
 journal      = {{TGDK}},
 volume       = {2},
 number       = {1},
 pages        = {3:1--3:37},
 year         = {2024},
 url          = {https://doi.org/10.4230/TGDK.2.1.3},
 doi          = {10.4230/TGDK.2.1.3},
}
```

Contents

1 Introduction

The vision of the Semantic Web as coined by Tim Berners-Lee, James Hendler, and Orla Lassila [22] in 2001 is to develop intelligent agents that can automatically gather semantic information from distributed sources accessible over the Web, integrate that knowledge, use automated reasoning [87], and solve complex tasks as such as schedule appointments in negotiation of the preferences of the involved parties. We have come a long way since then. In this paper, we reflect on the *past*, i. e., the ideas and components developed in the early days of the Semantic Web. Since the beginning, the Semantic Web has tremendously developed and undergone multiple waves of innovation. The Linked Data movement has especially seen uptake by industries, governments, and non-profit organizations, alike. We discuss those *present* components and concepts that have been added over the years and shown to be very useful. Although many concepts of the initial idea of the Semantic Web have been implemented and put into practice, still further research is needed to reach the full vision. Thus, this paper concludes with an outlook to *future* directions and steps that may be taken.

For the novice reader of the Semantic Web, we provide a brief historical overview of the developments and innovation waves of the Semantic Web: At the beginning of the Semantic Web, we were mainly talking about publishing Linked Data on the Web [100], i. e., semantic data typically structured using the Resource Description Framework (RDF)[1] that is accessible on the Web using URIs/IRIs to identify entities, classes, predicates, etc. By referencing entities from other websites and Web-accessible sources, i. e., dereferencable via HTTP, the data becomes naturally linked. By using standardized vocabularies and ontologies the information then becomes more aligned and easier to use across sources. These principles have allowed non-profit organizations, companies, governments, and individuals to publish and share large amounts of interlinked data, which has led to the success of the Linked Open Data cloud[2] since 2007. Since many of the large interconnected semantic sources are accessible via interfaces understanding structured query languages (SPARQL endpoints), federated query processing methods were developed that allow exploiting the strengths of structured query languages to precisely formulate an information need and optimize the query for efficient execution in a distributed setting.

When Google launched its Knowledge Graph in 2012[3], semantic technologies experienced another wave of new applications in the context of searching information. Whereas search engines before mainly relied on keyword search and string-based matches of the keywords in the websites' text, the knowledge graph enabled including semantics to capture the user's information need as well as the meaning of potentially relevant documents. To achieve this purpose, Google's knowledge graph integrates large amounts of machine-processable data available on the Web and uses this information not only to improve search results but also to display infoboxes for entities identified in the user's keywords. It is only since 2012 that we have widely used the term "knowledge graph" to refer to semantic data, where entities are connected via relationships and form large graphs of interconnected information, typically with RDF as a common standard language. In recent years though (labeled) property graphs (LPG) have been used to manage knowledge graphs. We refer to the literature for a detailed comparison of RDF graphs and LPGs [107] and also like to point out that they can be converted into each other [27]. In this article, we consider knowledge graphs from the perspective of the Semantic Web, i. e., we consider RDF graphs.

[1] http://www.w3.org/TR/rdf-primer/
[2] https://lod-cloud.net/
[3] https://blog.google/products/search/introducing-knowledge-graph-things-not/

A few years later, semantic technologies found another novel application in enterprise settings as enterprise knowledge graphs [77, 155] and data integration [42, 218]. Since all kinds of information can be structured as a graph, knowledge graphs can be used as a common structure to integrate heterogeneous information that is otherwise locked up in silos in different branches of a company. Integrating the data into a knowledge graph then allows for an integrated view, efficiently retrieving relevant information from this view once needed, and integrating external information that is available in the form of knowledge graphs, for instance on the Semantic Web. After all, online analytical processing-style queries can also be formulated with SPARQL[4] and evaluated on (distributed) knowledge graphs.

In the past few years, learning on graph data became one of the fastest growing and most active areas in machine learning [97]. Graph representation learning has created a new wave of graph embedding models and graph neural networks on knowledge graphs for tasks such as entity classification and link prediction. Natural Language Processing (NLP) has been another important field of the Semantic Web since the early years to extract knowledge from textual data and make it machine readable. Another field where NLP meets the Semantic Web is user interfaces for search on structured data to enable intuitive, natural language querying for graph data [95] similar to web search engines. At the end of 2022, ChatGPT[5] emerged as the first publicly available end-consumer tool based on a Large Language Model (LLM). Since then, the GPT-based family of LLMs has stirred up research and business alike and demonstrated impressive performance on many NLP tasks, including generating structured queries from user prompts and extracting structured knowledge from text [201].

The capabilities of tools like Bing Chat[6] with underlying access to the World Wide Web are reminding one of the intelligent agents that were envisioned 20 years before. For example, at the time of writing, the GPT4-based tool Bing[7] can internally generate SPARQL queries and execute them, while the structured response is seamlessly embedded into its natural language outputs to the users.[8] While it already addresses some of the early visions of the Semantic Web, particularly the complex planning and reasoning capabilities of LLMs are – due to their nature of focusing on generating and processing text – still limited. We hypothesize that advances in neuro-symbolic AI and semantic technologies will be key for improving LLMs and bringing generative AI tools like Bing and the Semantic Web further together. We are keen to witness this next era of the Semantic Web.

In this paper, we provide a comprehensive overview of the Semantic Web with its semantic technologies and underlying principles that have been inspiring and driving the multiple waves of innovations in the past two decades. Section 2 provides a motivating example for the classic Semantic Web. We refer back to this example throughout the paper. Section 3 presents the principles and the general architecture of the Semantic Web along with the basic semantic technologies it is founded upon. Besides classical components, we are also describing recent developments, and pointing out components that are still being researched and developed. Section 4 shows how to represent distributed knowledge on the Semantic Web. The creation and maintenance of graph data is described in Section 5. Section 6 discusses the principle of reasoning and logical inference. Section 7 then shows how to query over the (distributed) graph data on the Semantic

[4] http://www.w3.org/TR/sparql11-query/
[5] https://chat.openai.com/
[6] http://bing.com
[7] https://www.bing.com/
[8] Based on a sequence of prompts ran on January 22, 2024, using the GPT-4 model provided on the Bing Chat mobile app. The prompt sequence is: “do you have access to DBpedia”, “how do you access DBpedia”, “please give me an example where you access DBpedia in response”.

Web. We discuss the trustworthiness and provenance of data on the Semantic Web in Section 8. Machine learning methods for knowledge graphs are discussed in Section 9, particularly shallow graph embeddings and neural networks for graphs. We discuss language models and their relation and use for knowledge graphs in Section 10. We provide extensive examples of applications based on and using the Semantic Web and its technologies in Section 11. Finally, we reflect on the impact the Semantic Web has on practitioners in Section 12. Finally, we conclude with a brief outlook on future developments for the Semantic Web.

2 Motivating Example

On the Semantic Web, knowledge components from different sources can be intelligently integrated with each other. As a result, complex questions can be answered, questions like "What types of music are played on British radio stations? At which time and day of the week?" or "Which radio station plays songs by Swedish artists?" In this section, we provide an overview of how the Semantic Web can be employed to answer those questions. We provide details of the components of the Semantic Web in the following sections.

We consider the example of the BBC program ontology with links to various other ontologies such as for music, events, and social networks as shown in Figure 1. We start with the BBC playlists of its radio stations. The playlists are published online in Semantic Web formats. We can leverage the playlist to get unique identifiers of played artists and bands. For example, the music group "ABBA" has a unique identifier in the form of a URI (`https://www.bbc.co.uk/programmes/b03lyzpr`). This URI can be used to link the music group to information from the MusicBrainz[9] music portal. MusicBrainz knows the members of the band, such as Benny Andersson, as well as the genre and songs. In addition, MusicBrainz is linked to Wikipedia[10] (not shown in the figure), e. g., to provide information about artists, such as biographies on DBpedia [12]. Information about British radio stations can be found in the form of lists on Web pages such as Radio UK[11], which can also be converted into a representation in the Semantic Web.

We can see that the required information is distributed across multiple knowledge components, e. g., BBC Program, MusicBrainz, and others. Each knowledge component can in principle provide different access to the data and utilize various ways to describe the data. Consequently, to answer the questions the data must be integrated. On the Semantic Web, data integration relies on ontologies describing data and the meaning of relations in data.

Colloquially, an ontology is a description of concepts and their relationships. Ontologies are used to formally represent knowledge on the Semantic Web.[12] For example, Dublin Core[13] provides a metadata schema for describing common properties of objects, such as the creator of the information, type, date, title, usage rights, and so on. Figure 1 presents ontologies used to describe data in our example. For example, the Playcount ontology[14] of the BBC is used to model which artist was played and how many times in the programs. Ontologies can be interconnected in the Semantic Web. For example, the MusicBrainz ontology is connected to the BBC ontology using the Playcount ontology. Different ontologies with varying degrees of formality and different relationships to each other are used by the BBC to describe their data (see also [173]).

[9] `http://musicbrainz.org/`
[10] `https://www.wikipedia.org/`
[11] `https://www.radio-uk.co.uk`
[12] An ontology definition is provided in Section 4.2.
[13] `https://www.dublincore.org/specifications/dublin-core/dc-rdf/`
[14] `http://dbtune.org/bbc/playcount/`

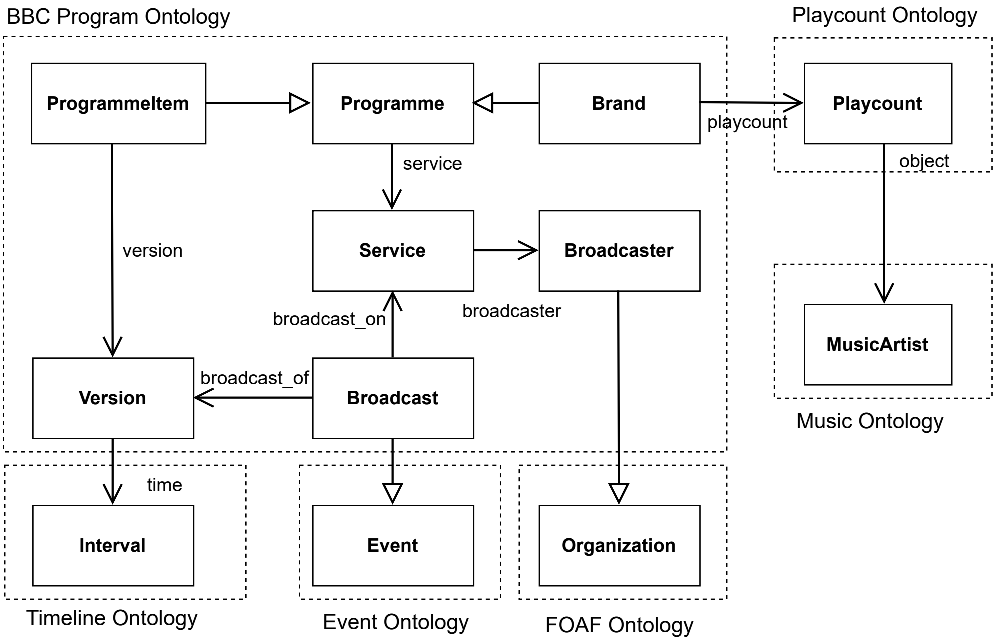


Figure 1 Example of the BBC ontology with links to other ontologies (notation based on UML, here without prefix for namespaces).

As all this data is available and interconnected by ontologies, a user of the Semantic Web can directly ask for answers to questions in this and other scenarios. To make this possible, the Semantic Web requires generic software components, languages, and protocols that can interact seamlessly with each other. We introduce the classical and modern components of the architecture of the Semantic Web in Section 3.

In addition to the above example, the Semantic Web can be used for a variety of other applications (see examples in Section 11). Apart from technical aspects, the Semantic Web should also be understood as a socio-political phenomenon. Similar to the World Wide Web, various individuals and organizations publish their data on the Semantic Web and collaborate to link and improve this data. This impact on practitioners is discussed in Section 12.

3 Architecture of the Semantic Web

The example in Section 2 describes *what* the Semantic Web is as an infrastructure, but not *how* this is achieved. In fact, the capabilities of the Semantic Web *in a small scale* have already been implemented by some knowledge-based systems originating from artificial intelligence research, e. g., Heinsohn et al [104]. However, for the implementation of the vision *on a large scale*, i. e, the Web, these knowledge-based systems lacked flexibility, robustness, and scalability. In part, this was due to the complexity of the algorithms used. For example, knowledge bases in description logic in the 1990s, which serve as the basis of Web ontologies, were limited regarding their size such that they could handle at the most some hundred concepts [104].

In the meantime, enormous improvements have been achieved. Greatly increased computational power and optimized algorithms allow a practical handling of large ontologies like Simple Knowledge

Organization System (SKOS)[15], Gene Ontology[16], Schema.org, and SNOMED-CT[17]. However, there are some fundamental differences between traditional knowledge-based systems and the Semantic Web. Data management in traditional knowledge-based systems has weaknesses in terms of handling large amounts of data and data sources, among other things because of different underlying formalisms, distributed locations, different authorities, different data quality, and a high frequency of change in the data used.

The Semantic Web applies fundamental principles to deal with these problems; they represent the basis for the architecture of the Semantic Web. This architecture's building blocks can roughly be categorized into groups, covering the entire life cycle of handling and managing graph data on the Web. These groups are graph data representation, creation and validation of graph data, reasoning over and linking of graph data, (distributed) querying of graph data, crypto, provenance, and trustworthiness of graph data, and user interfaces and applications.

Below, we first introduce the principles of the Semantic Web, from which we derive the architecture and its main components. Subsequently, we describe the groups of the architecture. The principles of the Semantic Web are:

1. *Explicit and simple data representation:* A general data representation abstracts from the underlying formats and captures only the essentials.
2. *Distributed systems:* A distributed system operates on a large set of data sources without centralized control that regulates which information belongs where and to whom.
3. *Cross-references:* The advantages of a network of data in answering queries are not based solely on the sheer quantities of data but on their interconnection, which allows reusing data and data definitions from other sources.
4. *Loose coupling with common language constructs:* The World Wide Web and likewise the Semantic Web are mega-systems, i. e., systems consisting of many subsystems, which are themselves large and complex. In such a mega-system, individual components must be loosely coupled in order to achieve the greatest possible flexibility. Communication between the components is based on standardized protocols and languages, whereby these can be individually adapted to specific systems.
5. *Easy publishing and easy consumption:* Especially in a mega-system, participation, i. e., publishing and consumption of data, must be as simple as possible.

These principles are achieved through a mix of protocols, language definitions, and software components. Some of these components have already been standardized by the W3C, which has defined both syntax and formal semantics of languages and protocols. Other components are not yet standardized, but they are already provided for the so-called *Semantic Web Layer Cake* by Tim Berners-Lee (cf. `http://www.w3.org/2007/03/layerCake.png`). We present a variant of the Semantic Web architecture, distinguishing between standardized languages and current developments. A graphical representation of the architecture can be found in Figure 2.

Identifier for Resources: HTTP, URL, DID

Entities (also called resources) are identified on the Internet by so-called Uniform Resource Identifiers (URIs) [20]. When a URI holds a dereferenceable location of the resource, in other words, it can be employed to get access to the resource via HTTP, it is called a Uniform Resource Locator (URL) [23, 21]. Furthermore, Internationalized Resource Identifiers (IRIs) [61] supplement

[15] `https://www.w3.org/TR/skos-reference/`
[16] `https://geneontology.org/`
[17] `https://www.snomed.org/`

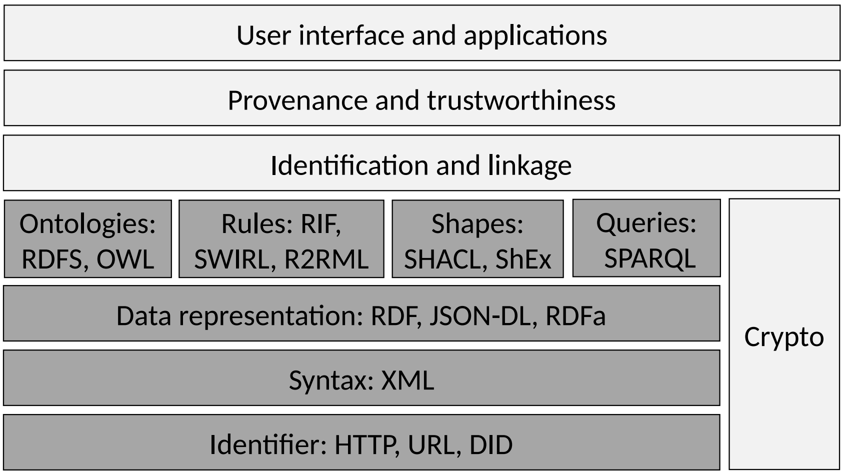

User interface and applications
Provenance and trustworthiness
Identification and linkage
Ontologies: RDFS, OWL
Rules: RIF, SWIRL, R2RML
Shapes: SHACL, ShEx
Queries: SPARQL
Data representation: RDF, JSON-DL, RDFa
Syntax: XML
Identifier: HTTP, URL, DID
Crypto

Figure 2 Representation of the components of the so-called "Semantic Web Layer Cake". W3C language standards are shown in dark gray. Current developments are shown in light gray.

URIs with international character sets from Unicode/ISO10646. URIs are globally and universally used but are usually not under our control. A recent W3C recommendation, the Decentralized Identifiers (DIDs), introduces an alternative approach to the above identifiers [192]. A DID is by default decentralized and allows for self-sovereign management of the identity, i. e., the control of a DID and its associated data is with the users.

In our example in Section 2, a URI [18] describes the musician Benny Andersson of the Swedish pop group ABBA. A user can dereference a URI that refers to ABBA, e. g., by performing a so-called look-up using HTTP to obtain a detailed description of the URI. We refer to the referenced standards for details.

For a detailed discussion of the role of dereferenceable URIs on the Semantic Web, we refer to the Linked Data principles described in Section 4.1.

Syntax for Data Exchange: XML, JSON-LD, RDFa

The Extensible Markup Language (XML)[19] is used to structure documents and enables the specification and serialization of structured data. In addition, other data formats were introduced to facilitate for serialization of RDF data, often replacing XML. We can view those formats as forming two groups. The first group consists of formats designed specifically for RDF data, such as Turtle[20], N-triple[21], and TRIG[22]. These are easier to view in a text editor, compared to XML, and thus easier to understand and modify. While initially not included in standards, their popularity has led to them being official W3C recommendations since 2014. The other group of formats is built by extending existing data formats. As a result, those can be employed to add RDF to

[18] http://www.bbc.co.uk/music/artists/2f031686-3f01-4f33-a4fc-fb3944532efa#artist
[19] https://www.w3.org/TR/xml/
[20] https://www.w3.org/TR/turtle/
[21] https://www.w3.org/TR/n-triples/
[22] https://www.w3.org/TR/trig/

existing systems. Examples of such formats are JSON-LD[23], CSV on the Web (CSVW)[24], and RDFa[25] extending JSON, CSV, and (X)HTML, respectively.

Graph Data Representation: RDF

In addition to the referencing of resources and a uniform syntax for the exchange of data, a data model is required that allows resources to be described both individually and in their entirety and how they are linked [100, 103]. An integrated representation of data from multiple sources is provided by a data model based on directed graphs [162]. The corresponding W3C standard language is RDF (Resource Description Framework)[26].

An RDF graph consists of a set of RDF triples, where a triple consists of a subject, predicate (property), and object. An RDF graph can be given an identifier, such a graph is called a named graph [38]. RDF graphs can be serialized in several ways (see Syntax for Data Exchange above). It is important to note that some formats (e. g., Turtle) do not support named graphs. Finally, RDF-star (aka RDF*)[27,28] was introduced to allow nesting of triples and thus enables an efficient way to make statements about statements while avoiding reification and the increased number of triples and complexity that come along with it.

The representation of graph data is further discussed in Section 4.

Creation and Validation of Graph Data: RIF, SWRL, [R2]RML, and SHACL

In the RDF context, a rule is a logical statement employed to infer new facts from existing graph data or to validate the data itself. RIF (Rule Interchange Format)[29] is a W3C recommendation format designed to facilitate the seamless interchange of rules between different rule engines. This enables the extraction of rules from one engine, their translation into RIF, publication, and subsequent conversion into the native syntax of another rule engine for execution. SWRL[30] is a rule-based language designed for representing complex relationships and reasoning.

Rules can also be used to state the correspondence between data sources and RDF graphs. The RDB to RDF Mapping Language (R2RML)[31] and the RDF Mapping Language (RML)[32] correspond to rule-based mapping languages for the declarative definition of RDF graphs. R2RML is the W3C recommendation for representing mappings from relational databases to RDF datasets, while RML extends R2RML to express rules not only from relational databases but also from data in the format of CSV, JSON, or XML.

Validating constraints, representing syntactic and semantic restrictions in RDF graphs, is essential for ensuring data quality. In addition to rule-based languages, shapes allow for the specification of conditions to meet data quality criteria and integrity constraints. A shape encompasses a conjunction of constraints representing conditions that nodes in an RDF graph must satisfy [108]. A shapes graph is a labeled directed graph where nodes correspond to shapes, and edges denote interrelated constraints. The Shapes Constraint Language (SHACL) [129] and

[23] https://www.w3.org/TR/json-ld/
[24] https://www.w3.org/TR/tabular-data-primer/
[25] https://www.w3.org/TR/rdfa-primer/
[26] https://www.w3.org/RDF/
[27] https://w3c.github.io/rdf-star/
[28] https://blog.liu.se/olafhartig/2019/01/10/position-statement-rdf-star-and-sparql-star/
[29] http://www.w3.org/2005/rules/wiki/RIF_Working_Group
[30] https://www.w3.org/submissions/SWRL/
[31] https://www.w3.org/TR/r2rml/
[32] https://rml.io/specs/rml/

Shape Expressions (ShEx) [167]) are two W3C-recommendations to express shapes graphs over RDF [169].

The creation and validation of graph data are described in detail in Section 5. We introduce OWL and reasoning on graph data below.

Reasoning and Linking of Graph Data: RDFS, OWL

Data from different sources may be heterogeneous. In order to deal with this heterogeneity and to model the semantic relationships between resources, the RDF Schema (RDFS)[33] vocabulary extends RDF by modeling types of resources (so-called RDF classes) and semantic relationships on types and properties in the form of generalizations and specializations. Likewise, it can be used to model the domain and range of properties.

RDFS is not expressive enough to merge data from different sources and define consistency criteria about it, such as the disjointness of classes or the equivalence of resources. The Web Ontology Language (OWL)[34] [1] is an ontology language with formally defined meaning based on description logic. This allows for reasoning services to be provided by knowledge-based systems for OWL ontologies. OWL can be exchanged using RDF data formats. Compared to RDFS, OWL provides more expressive language constructs. For example, OWL allows the specification of equivalences between classes and cardinality constraints on properties [1].

Reasoning over RDF graphs, incorporating RDFS and OWL models enhances semantic expressiveness and inferential capabilities. This involves making implicit information explicit, inferring new triples, and validating the RDF graph's consistency against defined ontological constraints. RDFS provides basic entailment regimens, creating hierarchies and simple inferencing via sub-class and sub-property relationships. In contrast, OWL introduces advanced constructs such as property characteristics (e. g., functional, inverse, symmetric properties), cardinalities, and disjointness axioms, enabling more expressive and complex modeling. Integrating RDFS and OWL reasoning mechanisms empowers applications to derive insights, discover implicit knowledge, and ensure adherence to specified ontological constraints within RDF-based knowledge representations.

Graph data aggregated from many data sources, such as in our example in Section 2, may contain many different identities. But those identities may represent the same set of real-world objects. Integration and linkage mechanisms allow references to be made between data from different sources. A popular approach to state the identity of two resources v and w is the `owl:sameAs` feature of OWL.

We discuss the reasoning over and linking of graph data in Section 6.

Querying of Graph Data: SPARQL

Since RDF makes it possible to integrate data from different sources, a query language is needed that allows formulating queries over individual RDF graphs as well as over the combination of multiple RDF graphs across multiple sources. SPARQL[35] (a recursive acronym for SPARQL Protocol and RDF Query Language) is a declarative query language for RDF graphs that enables us to formulate such queries. SPARQL 1.1[36] is the current version of SPARQL, which includes the capability to formulate federated queries over distributed data sources.

[33] `https://www.w3.org/TR/rdf11-schema/`

[34] `https://www.w3.org/OWL/`

[35] `http://www.w3.org/TR/rdf-sparql-query/`

[36] `http://www.w3.org/TR/sparql11-query/`

The basic building blocks of a SPARQL query are triple and graph patterns. A triple pattern corresponds to an RDF triple but where one, two, or all three of its components are replaced by variables (denoted with a leading "?"). These triple patterns with variables are to be matched in the queried graph. Multiple triple patterns can be combined into more complex graph patterns describing the connections between multiple nodes in the graph. The solution to such a SPARQL query then corresponds to all the subgraphs in an RDF graph matching this pattern.

Finally, there is RDF-star (aka RDF*)[37,38] – along with the corresponding SPARQL-star/ SPARQL* extension – was proposed and since then was implemented by several triple stores [3] that often provide publicly accessible SPARQL endpoints. The key idea with RDF/SPARQL-star is to allow the nesting of triples to enable an efficient way to allow statements about statements while avoiding reification and the increased number of triples and complexity that come along with it.

We describe SPARQL and federated querying in Section 7.

Crypto, Provenance, and Trustworthiness of Graph Data

Other aspects of the Semantic Web are encryption and authentication to ensure that data transmissions cannot be intercepted, read, or modified. Crypto modules, such as SSL (Secure Socket Layer), verify digital certificates and enable data protection and authentication. In addition, there are digital signatures for graphs that integrate seamlessly into the architecture of the Semantic Web and are themselves modeled as graphs again [125]. This allows graph signatures to be applied iteratively and enables to building trust networks. The Verifiable Credentials Data Model, a recent W3C recommendation, introduces a standard to model trustworthy credentials for graphs on the web[39]. Data on the Semantic Web can be augmented with additional information about its trustworthiness and provenance.

Aspects of trustworthiness and provenance of graph data as well as crypto are discussed in Section 8.

User Interfaces and Applications

A user interface enables users to interact with data on the Semantic Web. From a functional perspective, some user interfaces are generic and operate on the graph structure of the data, whereas others are tailored to specific tasks, applications, or ontologies. New paradigms are exploring the spectrum of possible user interfaces between generality and specific end-user requirements.

Semantic Web applications are discussed in Section 11. The impact on practitioners is described in Section 12.

4 Representation of Graph Data

The Linked Open Data principles are notably the most successful and widely adopted choice for representing RDF graph data on the web. Thus, we first introduce the reader to how to represent graph data as Linked Data. Subsequently, we introduce the notion of ontologies. This is followed by a more detailed analysis of the different types of ontologies. We give examples of ontologies throughout the sections. With this background in mind, we reconsider our running example

[37] https://w3c.github.io/rdf-star/

[38] https://blog.liu.se/olafhartig/2019/01/10/position-statement-rdf-star-and-sparql-star/

[39] https://www.w3.org/TR/vc-data-model/

from Section 2 and analyze the given distributed network of ontologies. In this context, we also introduce and discuss the notion of ontology design patterns.

4.1 Linked Graph Data on the Web

The *Linked Data* principles[40] define the methods for representing, publishing, and using data on the Semantic Web. They can be summarized as follows:

1. URIs are used as names for entities.
2. The HTTP protocol's GET method is used to retrieve descriptions for a URI.
3. Data providers shall return relevant information in response to HTTP GET requests on URIs using standards, e. g., in RDF.
4. Links to other URIs shall be used to facilitate knowledge discovery and use of additional information.

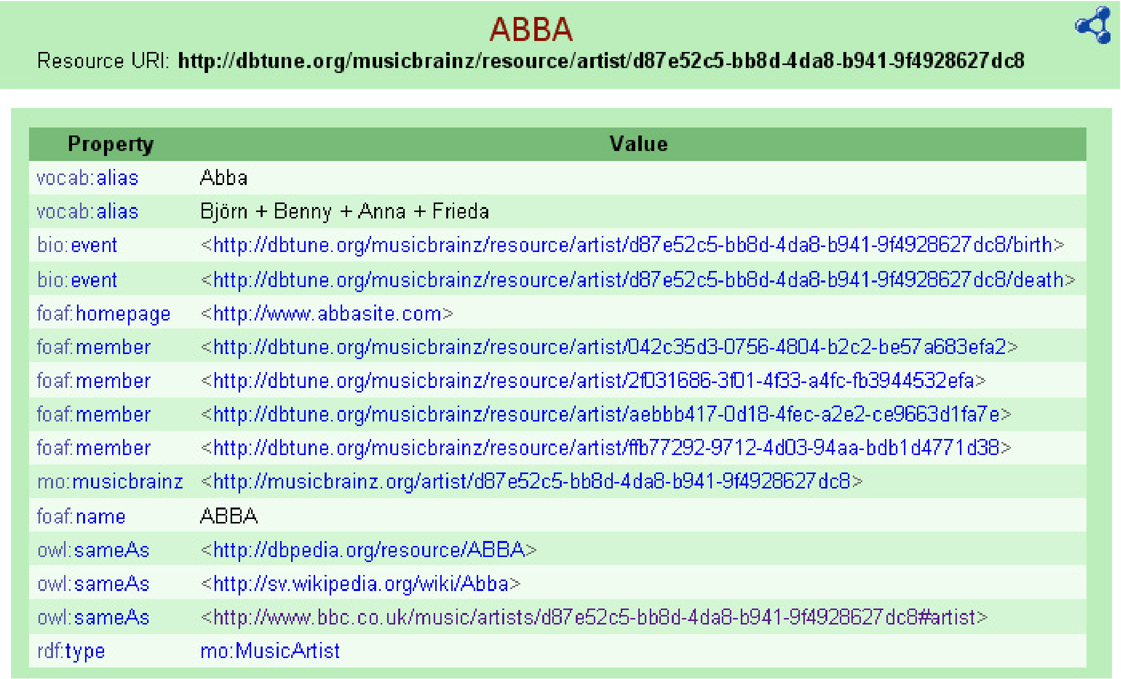
Publishing data using Linked Data principles allows easy access to data via HTTP. This allows exploration of resources and navigation across resources on the Semantic Web. URIs (see 1.) are dereferenced using HTTP requests (2.) to obtain additional information about a given resource. In particular, using standardized syntax (3.), this information may also contain links to other resources (4.).

Figure 3 represents an example of Linked Data about the pop group ABBA. The example describes several relationships linking entities to ABBA's URI, such as `foaf:member` and `rdf:type`. In the figure "ABBA", or more precisely the URI of ABBA, is the subject, "Property" refers to relationships, and "Value" represents objects of the RDF triples. The relation `owl:sameAs` will be explained in more in Section 6. The prefixes `foaf`, `rdf`, and `owl` refer to vocabularies of the FOAF ontology[41], and the W3C language specifications of RDF and OWL, respectively.

4.2 Ontologies

An ontology is commonly defined as a formal, machine-readable representation of key concepts and relationships within a specific domain [158, 156]. In essence, ontologies capture a shared perspective [158] that is, the formal conceptualization of ontologies expresses a consensus view among different stakeholders. Visualizing ontologies is akin to viewing a spectrum, with a specificity of concepts, their relationships, and the granularity of meaning representation varying along this continuum [142, 205, 204]. A controlled vocabulary corresponds to the less expressive form of ontology, comprising a restrictive list of words or terms used for labeling, indexing, or categorization. The Clinical Data Interchange Standards Consortium (CDISC) Terminology is an exemplary vocabulary that harmonizes definitions in clinical research.[42] A thesaurus is located next in the spectrum; they enhance controlled vocabularies with information about terms and their synonyms and broader/narrower relationships. The Unified Medical Language System (UMLS) integrates medical terms and their synonyms.[43] Next, taxonomies are built over controlled vocabularies to provide a hierarchical structure, e. g., parent/child relationship. SNOMED-CT[44] (Systematized Nomenclature of Medicine Clinical Terms) provides a terminology and coding system used in healthcare and medical fields; medical concepts organized in a hierarchical structure enabling a granular representation of clinical information. The Simple Knowledge Organization

[40] `http://www.w3.org/DesignIssues/LinkedData.html`
[41] `http://xmlns.com/foaf/spec/`
[42] `https://datascience.cancer.gov/resources/cancer-vocabulary/cdisc-terminology`
[43] `https://www.nlm.nih.gov/research/umls/index.html`
[44] `https://www.snomed.org/value-of-snomedct`



ABBA

Resource URI: **http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8**

Property	Value
vocab:alias	Abba
vocab:alias	Björn + Benny + Anna + Frieda
bio:event	<http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8/birth>
bio:event	<http://dbtune.org/musicbrainz/resource/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8/death>
foaf:homepage	<http://www.abbasite.com>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/042c35d3-0756-4804-b2c2-be57a683efa2>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/2f031686-3f01-4f33-a4fc-fb3944532efa>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/aebbb417-0d18-4fec-a2e2-ce9663d1fa7e>
foaf:member	<http://dbtune.org/musicbrainz/resource/artist/ffb77292-9712-4d03-94aa-bdb1d4771d38>
mo:musicbrainz	<http://musicbrainz.org/artist/d87e52c5-bb8d-4da8-b941-9f4928627dc8>
foaf:name	ABBA
owl:sameAs	<http://dbpedia.org/resource/ABBA>
owl:sameAs	<http://sv.wikipedia.org/wiki/Abba>
owl:sameAs	<http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist>
rdf:type	mo:MusicArtist

Figure 3 Linked Data example for ABBA.

System (SKOS)[45] is a W3C standard to describe knowledge about organizational systems. Lastly, ontologies are at the highest extreme of the spectrum, integrating sets of concepts with attributes and relationships to define a domain of knowledge.

Note, SKOS is a popular standard for modeling domain-specific taxonomies in the different scientific communities such as economics, social sciences, etc. to represent concepts and their relationships, most importantly narrower, broader, and related. However, it does not have the expressiveness of OWL with its complex expressions on classes and relations. For a detailed discussion, we refer to the literature such as [121] and the W3C on using OWL and SKOS[46].

4.3 Types and Examples of Ontologies

A network of ontologies, such as the example shown in Figure 1, may consist of a variety of ontologies created by different actors and communities. Ontologies may be the result of a transformation or reengineering activity of a legacy system, such as a relational database or existing taxonomy such as the Dewey Decimal Classification[47] or Dublin Core. Other ontologies are created from scratch. This involves applying existing methods and tools for ontology engineering and choosing an appropriate representation language for the ontology (see Section 6).

Ontology engineering deals with the methods for creating ontologies [88] and has its origins in software engineering in the creation of domain models and in database design in the creation of conceptual models. A good overview of ontology engineering can be found in several reference books [88]. Ontologies vary greatly in their structure, size, development methods applied, and

[45] `https://www.w3.org/TR/skos-reference/`
[46] `https://www.w3.org/2006/07/SWD/SKOS/skos-and-owl/master.html`
[47] `https://www.oclc.org/en/dewey.html`

application domains considered. Complex ontologies are also distinguished in terms of their purpose and granularity.

Domain Ontologies represent knowledge specific to a particular domain [65, 156]. Domain ontologies are used as external sources of background knowledge [65]. They can be built on foundational ontologies [157] or core ontologies [184], which provide precise structuring to the domain ontology and thus improve interoperability between different domain ontologies. Domain ontologies can be simple such as the FOAF ontology or the event ontology mentioned above, or very complex and extensive, having been developed by domain experts, such as the SNOMED medical ontology.

Core Ontologies represent a precise definition of structured knowledge in a particular domain spanning multiple application domains [184, 156]. Examples of core ontologies include core ontologies for software components and web services [156], for events and event relationships [182], or for multimedia metadata [178]. Core ontologies should thereby build on foundational ontologies to benefit from their formalization and strong axiomatization [184]. For this purpose, new concepts and relations are added to core ontologies for the application domain under consideration and are specialized by foundational ontologies.

Foundational Ontologies have a very wide scope and can be reused in a wide variety of modeling scenarios [32]. They are therefore used for reference purposes [156] and aim to model the most general and generic concepts and relations that can be used to describe almost any aspect of our world [32, 156], such as objects and events. An example is the Descriptive Ontology for Linguistic and Cognitive Engineering (DOLCE) [32]. Such basic ontologies have a rich axiomatization that is important at the developmental stage of ontologies. They help ontology engineers to have a formal and internally consistent conceptualization of the world, which can be modeled and checked for consistency. For the use of foundational ontologies in a concrete application, i. e., during the runtime of an application, the rich axiomatization can often be removed and replaced by a more lightweight version of the foundational ontology.

In contrast, domain ontologies are built specifically to allow automatic reasoning at runtime. Therefore, when designing and developing ontologies, completeness and complexity on the one hand must always be balanced with the efficiency of reasoning mechanisms on the other. In order to represent structured knowledge, such as the scenario depicted in Figure 1, interconnected ontologies are needed, which are spanned in a network over the Internet. For this purpose, the ontologies used must match and be aligned with each other.

4.4 Distributed Network of Ontologies and Ontology Patterns

A network of ontologies must be flexible with respect to the functional requirements imposed on it. This is because systems are modified, extended, combined, or integrated over time. In addition, the networked ontologies must lead to a common understanding of the modeled domain. This common understanding can be achieved through a sufficient level of formalization and axiomatization, and through the use of ontology patterns. An ontology pattern, similar to a design pattern in software engineering, represents a generic solution to a recurring modeling problem [184]. Ontology patterns allow to select parts from the original ontology. Either all or only certain patterns of an ontology can be reused in the network. Thus, to create a network of ontologies, e. g., existing ontologies and ontology patterns can be merged on the Web. The ontology engineer can drive or explicitly provide for the modularization of ontologies using ontology patterns. Core ontologies represent one approach to designing a network of ontologies (see in detail [184]). They allow to capture and exchange structured knowledge in complex domains. Well-defined core ontologies fulfill the properties mentioned in the previous section and allow easy integration and smooth interaction of ontologies (see also [184]). The networked ontologies approach leads to a flat structure, as shown in Figure 1, where all ontologies are used on the same level. Such structures can be managed up to a certain level of complexity.

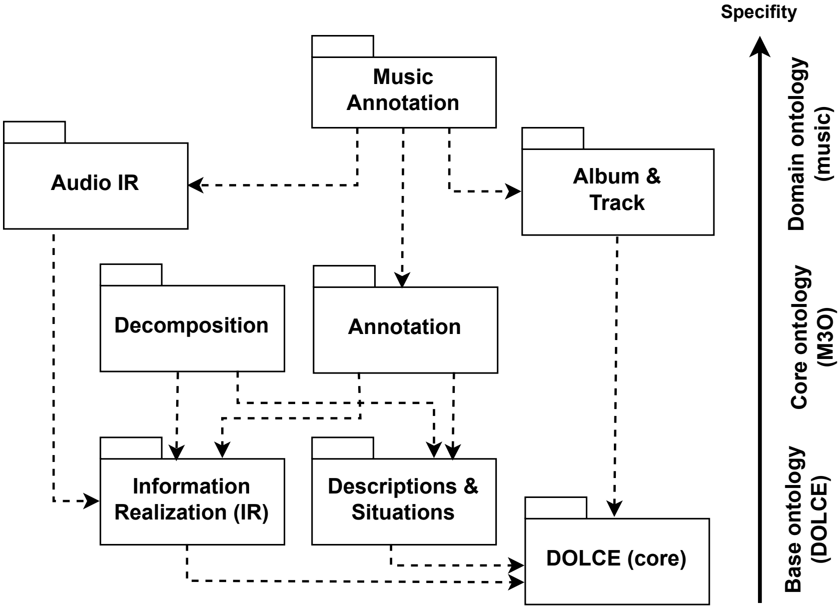


Figure 4 Ontology layers combining the foundational ontology DOLCE, the multimedia metadata ontology M3O, domain-specific extensions to M3O for annotating audio data and music, and a domain ontology for albums and tracks.

The approach of networked core ontologies is illustrated by the example of ontology layers starting from foundational to core to domain ontologies. As shown in Figure 4, DOLCE is the foundational ontology at the bottom layer, the Multimedia Metadata Ontology (M3O) [178] as the core ontology for multimedia metadata, and an extension of M3O for the music domain. Core ontologies are typically defined in description logic and cover a field larger than the specific application domain requires [79]. Concrete information systems will typically use only a subset of core ontologies. To achieve modularization of core ontologies, they should be designed using ontology patterns. By precisely matching the concepts in the core ontology with the concepts provided in the foundational ontology, they provide a solid foundation for future extensions. New patterns can be added and existing patterns can be extended by specializing the concepts and roles. Figure 4 shows different patterns of the M3O and DOLCE ontologies.

Ideally, the ontology patterns of the core ontologies are reused in the domain ontologies [79], as shown in Figure 4. However, since it cannot be assumed that all domain ontologies are aligned with a foundational or core ontology, the option that domain ontologies are developed and maintained independently must also be considered. In this case, domain knowledge can be reused in core ontologies by applying the Descriptions and Situations (DnS) ontology pattern of the foundational ontology DOLCE. The DnS ontology pattern is an ontological formalization of context [156] by defining different views using roles. These roles can refer to domain ontologies and allow a clear separation of the structured knowledge of the core ontology and domain-specific knowledge. To model a network of ontologies, such as the example described above, the Web Ontology Language (OWL) and its ability to axiomatize using description logic [15] is used. In addition to being used to model a distributed knowledge representation and integration, OWL, is also used in particular to derive inferences from this knowledge, which is described in Section 6.

5 Creation and Validation of Graph Data

In this section, we describe the creation of graph data from legacy data. Many tools are available for this task, which support various mappings and transformations. Subsequently, we discuss data quality and the validation of knowledge graphs, including the recent approaches on shapes. We also reflect on the role of the open-world versus closed-world assumption with respect to validating data.

5.1 Graph Data Creation

Graph data can be created by transforming legacy data via a data integration system [133], which consists of a unified schema, data sources, and mapping rules. These mapping rules define the concepts within the schema and establish links to the data sources. By employing declarative definitions, knowledge graph creation promotes modularity and reusability. This approach allows users to trace the entire graph creation process, leading to improved transparency and ease of maintenance.

To enable comprehensive and extensive graph specification, mappings and transformations have been developed to convert data from various storage models into Semantic Web data models like RDF. These mappings and transformations facilitate the mapping of data into RDF, thereby supporting the integration of diverse data sources into the Semantic Web.

The mapping language R2RML [51] defines mapping rules from relational databases (relational data models) to RDF graphs. These mappings themselves are also RDF triples [18]. Because of its compact representation, Turtle is considered a user-friendly notation of RDF graphs. The structure of R2RML is illustrated in Figure 5; essentially, table contents are mapped to triples by the classes `SubjectMap`, `PredicateMap`, and `ObjectMap`. If the object is a reference to another table, this reference is called `RefObjectMap`. Here, `SubjectMap` contains primary key attributes of the corresponding table. Thus, there exists a mapping rule representable in RDF graphs by means of which tables of relational databases can be represented as RDF graphs.

The RDF Mapping Language (RML)[57] extends R2RML to encompass the definition of logical sources in various formats, including CSV, JSON, XML, and HTML. This enhancement enables RML to introduce new operators that facilitate the integration of data from diverse sources into the Semantic Web. Thus, instead of `LogicalTable`, RML includes the tag `LogicalSource`, to allow for the retrieval of data in several formats. Additionally, RML resorts to W3C-standardized vocabularies and enables the definition of retrieval procedures to collect data from Web APIs or databases. R2RML and RDF mapping rules are expressed in RDF, and their graphs document how classes and properties in one or various ontologies that are part of an RDF graph are populated from data collected from potentially heterogeneous data sources.

Over time, the Semantic Web community has actively contributed to addressing the challenge of integrating heterogeneous datasets, resulting in the development of several frameworks for executing declarative mapping rules [43, 36, 166]. A rich spectrum of tools (e. g., RMLMapper [57], RocketRML [191], CARML[48], SDM-RDFizer [118], Morph-KGC [9], and RMLStreamer [159]) offers the possibility of executing R2RML and RML rules and efficiently materializing the transformed data into RDF graphs. Van Assche et al. [11] provided an extensive survey detailing the main characteristics of these engines. Despite significant efforts in developing these solutions, certain parameters can impact the performance of the graph creation process [40]. Existing engines may face challenges when handling complex mapping rules or large data sources. Nonetheless, the

[48] https://github.com/carml/carml

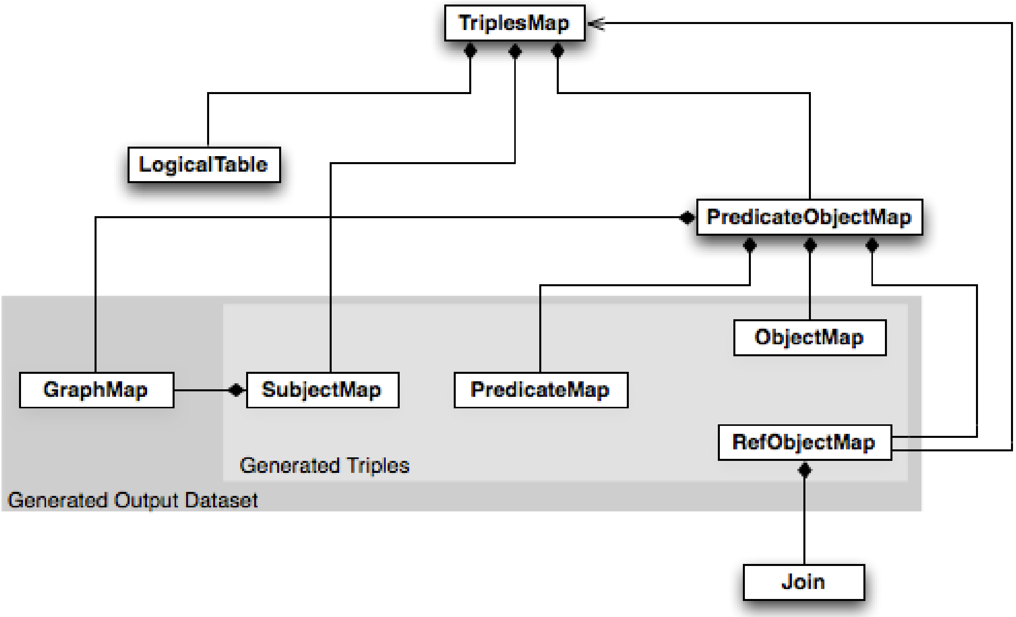


Figure 5 Structure of a relational data mapping (source: [51]).

community continues to collaborate and address these issues. An example of such collaboration is the Knowledge Graph Construction Workshop 2023 Challenge[49] that took place at ESWC 2023. This community event aims to understand the strengths and weaknesses of existing approaches and devise effective methods to overcome existing limitations.

RDF graphs can also be dynamically created through the execution of queries over data sources. These queries involve the rewriting of queries expressed in terms of an ontology, based on mapping rules that establish correspondences between data sources and the ontology. Tools such as Ontop [36], Ultrawrap [189], Morph [166], Squerall [141], and Morph-CSV [41] exemplify systems that facilitate the virtual creation of RDF graphs.

5.2 Quality and Validation of Graph Data

Quality and validation of the graph data are crucial to maintaining the integrity of the Semantic Web [59, 52, 226]. The evaluation of integrity constraints allows for the identification of inconsistencies, inaccuracies, or contradictions within the data. They also help maintain consistency by ensuring related data elements remain coherent. Constraints are logical statements – expressed in a particular language – that impose restrictions on the values taken for target nodes in a given property.

Constraints can be expressed using OWL [200], SPARQL queries [132], or using shapes. However, the interpretation of the results depends on the semantics followed to interpret the failure of an integrity constraint. For example, constraints expressed in OWL are validated using an Open-World Assumption (OWA) (i. e., a statement cannot be inferred to be false based on failures to prove it) and under the absence of the Unique Name Assumption (UNA) (i. e., two

[49] https://zenodo.org/record/7689310

different names may refer to the same object). These two features make it difficult to validate data in applications where data is supposed to be complete. Definitions of integrity constraint semantics in OWL using the Closed-World Assumption [149, 150, 200] overcome these issues.

Contrarily, constraints expressed using SPARQL queries or shapes will be evaluated under the Closed-World Assumption (CWA) and following the Unique Name Assumption (UNA). Nevertheless, some constraints may be difficult to express in SPARQL, and the specification process is prone to errors and difficult to maintain.

Data quality conditions and integrity constraints can also be expressed as graphs of shapes or the so-called *shapes schema*. A shape corresponds to a conjunction of constraints that a set of nodes in an RDF graph must satisfy [108]. These constraints can restrict the types of nodes, the cardinality of certain properties, and the expected data types or values for specific properties. A shape can target the instances of a class, the domain or range of a property, or a specific node in the RDF graph. A shape or node in a shapes graph is validated in an RDF graph, if and only if, all the target nodes in the RDF graph satisfy all the constraints in the shape. Figure 6 presents a shapes graph of three shapes targeting the classes `Brand`, `Playcount`, and `MusicArtist`. Each of the shapes comprises one constraint. In the shapes `Brand` and `MusicArtist` the properties title and name can take more than one value, while the shape `Playcount` states that each instance of the class `Playcount` must have exactly one value of the property `count`. Additionally, the instances of the class `Brand` must be related to valid instances of the class `Playcount` which should also be related to valid instances of the class `MusicArtist`.

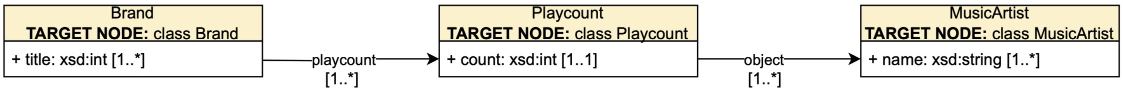


Figure 6 Shapes for Graph Data. A shapes graph comprises three shapes interlinked by the properties `playcount` and `objects` between the target classes `Brand`, `Playcount`, and `MusicArtist`.

There are two standards for defining shapes, ShEx (Shape Expressions) [167]) and SHACL (Shapes Constraint Language) [129]). Both define shapes over the attributes (i. e., `owl:Datatype-Properties`), and constraints on incoming/outgoing arcs, cardinalities, RDF syntax, and extension mechanism. These inter-class constraints induce a shape network used to validate the integrity and data quality properties of an RDF graph.

SHACL and ShEx, although sharing a common goal, adopt distinct approaches. ShEx seeks to offer a language serving as a grammar or schema for RDF graphs, delineating RDF graph structures for validation. On the other hand, SHACL is positioned as the W3C recommendation for validating RDF graphs against a conjunction of constraints, emphasizing a constraint language for RDF. Despite their analogous roles in specifying shapes and constraints for RDF data, ShEx and SHACL differ in syntax, expressiveness, and community adoption [81].

The evaluation results of a SHACL shape network over an RDF graph are presented in validation reports using a controlled vocabulary. A validation report includes explanations about the violations, the severity of the violation, and a message describing the violation. SHACL is the language selected by the International Data Space (IDS) to express the restrictions that state the integrity over RDF graphs [131]. Besides the integrity validation of an RDF graph, SHACL can be utilized to describe data sources and the certification of a query answer [176], as metadata to enhance the performance of a SPARQL query engine [168], to certify access policies [177], and to provide provenance as a result of the validation of integrity constraints [54].

In the context of a quality assessment pipeline, one crucial step involves validating the shape schema against a graph. It is important to mention that the validation of recursive shape schemas is not explicitly addressed in the SHACL specification [129]. To address this gap, Corman et al. [48] introduce a semantic framework for validating recursive SHACL. They also demonstrated that validating full SHACL features is an NP-hard problem. Building on these insights, they proposed specific fragments of SHACL that are computationally tractable, along with a fundamental algorithm for validating shape schemas using SPARQL [47]. In a related vein, Andresel et al. [8] propose a stricter semantics for recursive SHACL, drawing inspiration from stable models employed in Answer Set Programming (ASP). This innovative approach enables the representation of SHACL constraints as logic programs and leverages existing ASP solvers for shape schema validation. Importantly, this approach allows for the inclusion of negations in recursive validations. Further, Figuera et al. [70] present Trav-SHACL, an approach that focuses on query optimization techniques aimed at enhancing the incremental behavior and scalability of shape schema validation.

While SHACL has been adopted in a broad range of use cases, given a large graph it remains a challenge how to define shapes efficiently [169]. In many industrial settings with billions of entities and facts [155] creating shapes manually simply is not an option. The current state of the art can automatically extract shapes on WikiData (ca. 2 Billion facts) in less than 1.5 hours while filtering shapes based on the well-established notions of support and confidence to avoid reporting thousands of shapes that are so rare or apply to such a small subset of the data that they become meaningless [170]. Still, more work is needed to increase scalability further and also to help users make good use of the mined shapes [171] and, e. g., interactively use them to correct and improve the quality of their graphs.

6 Reasoning over and Linking of Graph Data

Section 3 introduced several formal languages for knowledge representation on the Semantic Web. RDF allows the description of simple facts (statements with subject, predicate, and object, so-called RDF triples), e. g., “Anni-Frid Lyngstad” “is a member of” “ABBA”. RDFS allows the definition of types of entities (classes), relationships between classes, and a subclass and superclass hierarchy between types (analogously for relations). OWL is even more expressive than RDF and RDFS. For example, OWL allows the definition of disjoint classes or the description of classes in terms of intersection, union, and complement of other classes.

Below, we first introduce the reasoning over RDFS and OWL at the example of our BBC scenario from Section 2. Subsequently, we discuss works on linking data objects and concepts.

6.1 Reasoning over Graph Data

Based on formal languages representing graph data and their semantics, further (implicit) facts can be derived from the knowledge base by deductive inference. In the following, we exemplify the derivation of implicit facts from a set of explicitly given facts using the RDFS construct `rdfs:subClassOf` and the OWL construct `owl:sameAs`. The property `rdfs:subClassOf` describes hierarchical relationships between classes and with `owl:sameAs` two resources can be defined as identical.

As a first example, we consider the class `foaf:Person`, which is defined in the FOAF ontology, and the classes `mo:Musician` and `mo:Group`, which are defined in the music ontology. In the music ontology, there is an additional axiom that defines `mo:Musician` as a subclass of `foaf:Person` using `rdfs:subClassOf`. Given this axiom, it can be deduced by deductive inference that instances of `mo:Musician` are also instances of `foaf:Person`. Now if there is such a hierarchy of classes and in addition a statement that Anni-Frid Lyngstad is of type `mo:Musician`, then it can be inferred by inference that Anni-Frid Lyngstad is also of type `foaf:Person`. This means that all queries asking for entities of type `foaf:Person` will also include Anni-Frid Lyngstad in the query result, even if that entity is not explicitly defined as an instance of foaf:Person. Figure 7 represents these facts and the corresponding class hierarchy in RDFS as a directed graph.

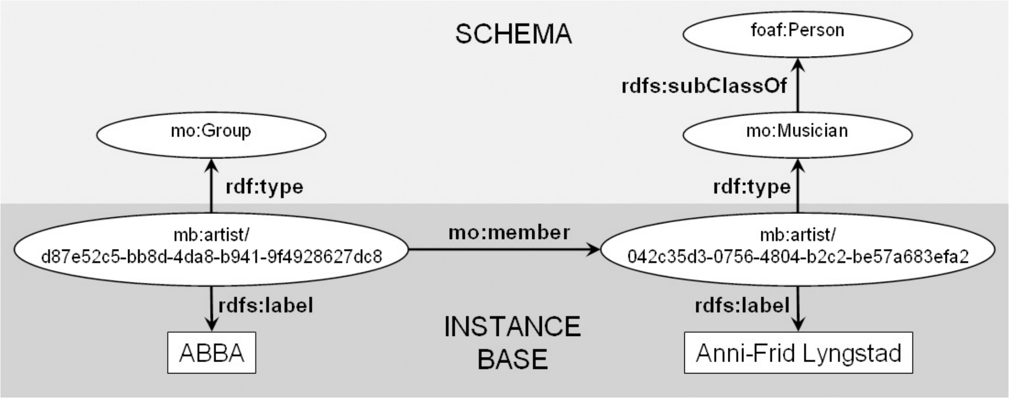


Figure 7 Visualization of RDF sample data about ABBA and Anni-Frid Lyngstad to illustrate inference in RDFS.

In the second example, the OWL construct owl:sameAs is used to define two resources as identical, for example http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist and http://dbpedia.org/resource/ABBA. Identical here means that these two URIs represent the same real-world object. By inference, information about ABBA from different sources can now be linked. Since ontologies are created independently on the web, and URIs are subject to local naming conventions, a real-world object may be represented by multiple URIs (in different ontologies).

OWL offers a variety of other constructs for the description of classes, relationships, and concrete facts. For example, OWL allows the declaration of transitive relations and inverse relations. For example, the relation "is-member" is inverse to "has-member". OWL reasoning allows, among other things, consistency checking of an ontology or checking the satisfiability of classes [109]. A class is satisfiable if there can be instances of that class.

For a detailed discussion about OWL reasoning, we refer to the literature such as [109, 16]. Different reasoners for OWL have seen widespread adoption in the community such as the well-known Pellet[50] and Hermit [85]. Finally, a combination of description logic and rules is also possible. For example, Motik et al. [151] presented a combination of description logic and rules that allows tractable inference on OWL ontologies.

6.2 Linking of Objects and Concepts

In the Semantic Web, it cannot be assumed that two URIs refer to two different real-world objects (cf. unique name assumption in Section 5.2). A URI by itself, or in itself, has no identity [96]. Rather, the identity or interpretation of a URI is revealed by the context in which it is used on the Semantic Web. Determining whether or not two URIs refer to the same entity is not a simple task and has been studied extensively in data mining and language understanding in the past. For example, to identify whether or not the author names of research papers refer to the same person, it is often not sufficient to resolve the name, venue, title, and co-authors [122]. The process of determining the identity of a resource is often referred to as entity resolution [122], coreference

[50] https://github.com/stardog-union/pellet

resolution [214], object identification [174], and normalization [214, 215]. Correctly determining the identity of entities on the Web is important as more and more records appear on the Web and this presents a significant hurdle for very large Semantic Web applications [84].

To address this, a number of services exist that can recognize entities and determine their identity: Thomson Reuters offers OpenCalais[51], a service that can link natural language text to other resources using entity recognition. Another commercial tool that allows for extracting knowledge graphs from text is provided by DiffBot.[52] Recently, the language model ChatGPT has been compared to the specialized entity and relation extraction tool REBEL [35] for the task of creating knowledge graphs from sustainability-related text [201]. The experiments suggest that large language models improve the accuracy of creating knowledge graphs [201]. The sameAs[53] service aims to detect duplicate resources on the Semantic Web using the OWL relationship `owl:sameAs`. This can be used to resolve coreferences between different datasets. For example, for the query with the URI `http://dbpedia.org/resource/ABBA`, a list of over 100 URIs is returned that also references the music group ABBA. One of them is BBC with the resource `http://www.bbc.co.uk/music/artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist`.

Furthermore, the problem of schema matching [215] is very related to the problem of entity resolution, co-reference resolution, and normalization. The goal of schema matching is to address the question of how to integrate data [215], which is non-trivial even for small schemas. In the Semantic Web, schema matching means the matching of different ontologies, respectively the concepts defined in these ontologies. Various (semi-)automatic or machine learning techniques for matching ontologies have been developed in the past [66, 62, 26]. Core ontologies as illustrated in Figure 4.3 represent generic modeling frameworks for integration and alignment with other ontologies. In addition, core ontologies can also integrate Linked Open Data, which typically contains no or very little schema information. The YAGO ontology [195] was generated from the fusion of Wikipedia and Wordnet using rule-based and heuristic methods. A manual assessment showed an accuracy of 95%.

Manual matching of different data sources is also pursued in the Linked Open Data project of the German National Library[54]. For example, the database containing the authors of all documents published in Germany was manually linked with DBpedia and other data sources. A particular challenge was to identify the authors, as described above. For example, former German Chancellor Helmut Kohl has a namesake whose work should not be linked to the chancellor's DBpedia entry. Relationships between keywords used to describe publications are asserted using the SKOS (Simple Knowledge Organization System) vocabulary.[55] For example, keywords are related to each other using the relation `skos:related`. Hyponyms and hypernyms are expressed by the relations `skos:narrower` and `skos:broader`. Finally, the Ontology Alignment Evaluation Initiative[56] should be mentioned, which aims to achieve an established consensus for evaluating ontology matching methods.

7 Querying of Linked Data

Queries over Linked Data can be processed using link traversal [101], i. e., the query processor would use one of those IRIs given directly in the query as starting point and query the respective

[51] `https://www.refinitiv.com/en/products/intelligent-tagging-text-analytics`
[52] `https://www.diffbot.com/`
[53] `https://www.sameas.cc/`
[54] `http://www.d-nb.de/`
[55] `https://www.w3.org/TR/2009/REC-skos-reference-20090818/`
[56] `http://oaei.ontologymatching.org/`

source for more triples involving the IRI. By iteratively doing this for more IRIs and with respect to the graph pattern defined in the query, a local set of triples is collected over which the given query can be evaluated.

More conveniently, queries over RDF and Linked Data can be formulated in SPARQL[57], if a corresponding endpoint to the graph data is made available. Whereas such queries can target graphs that are stored in a single graph store, Linked Data often requires formulating and executing queries across multiple graphs that are stored at distributed data sources.

Below, we first introduce the basic query processing of SPARQL queries along with our running example. This is followed by discussing RDFS/OWL entailment regimes and querying. Finally, we present approaches for distributed querying over multiple SPARQL endpoints.

7.1 Basic Query Processing

In principle, a SPARQL query is evaluated by comparing the graph pattern defined in the query to the RDF graph and reporting all matches as results. The set of results can be restricted by additional criteria, such as filters, i. e., conditions on variables and triple patterns that additionally need to be fulfilled.

As an example, let us consider the query illustrated in Figures 8 and 9 that we want to execute over our example MusicBrainz graph from Section 2. We are now interested in the musicians of ABBA who are also members of other bands. If we follow the Linked Data principles and evaluate the query using link traversal [101], this would mean first querying for triples including the IRI that represents ABBA, then navigating to the individual band members, and then following the links to all of the members' bands and query more relevant triples.

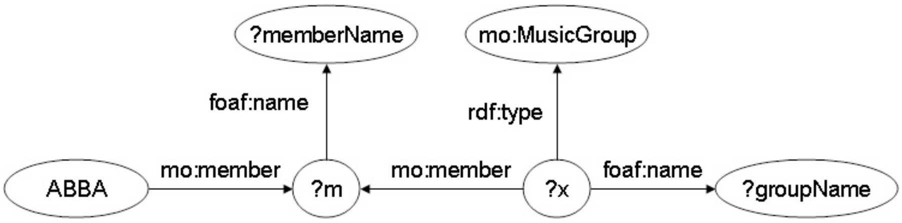


Figure 8 Graphical representation of a query for music groups (represented by the variable *?groupName*), whose members are also members of ABBA. The variable *?m* refers to the members of ABBA. The vertex labeled "ABBA" represents the URI for ABBA. The prefix *mo* refers to the music ontology, *foaf* to the FOAF ontology, and *rdf* to the vocabulary of the RDF specification.

Similar to relational database systems, there exist several dedicated graph stores (aka triple stores) that are optimized for RDF graphs and evaluating SPARQL queries. Some of the most popular triple stores are RDF4J [34], Jena [217], Virtuoso[58], and GraphDB[59]. They are building upon concepts and techniques known from relational database systems [112, 153] and expand them with graph-specific optimizations [211, 193, 94, 138, 69].

7.2 Entailment Regimes and Query Processing

In addition to explicitly querying existing facts, SPARQL provides inferencing support through so-called *entailment regimes*. They correspond to logical consequences describing the relationship

[57] `http://www.w3.org/TR/sparql11-query/`
[58] `http://virtuoso.openlinksw.com`
[59] `https://graphdb.ontotext.com/`

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX mo: <http://purl.org/ontology/mo/>
PREFIX foaf: <http://xmlns.com/foaf/0.1/>
PREFIX bbc: <http://www.bbc.co.uk/music/>
SELECT ?memberName ?groupName
WHERE  { bbc:artists/d87e52c5-bb8d-4da8-b941-9f4928627dc8#artist mo:member ?m .
         ?x mo:member ?m .
         ?x rdf:type mo:MusicGroup .
         ?m foaf:name ?memberName .
         ?x foaf:name ?groupName }
FILTER (?groupName <>  "ABBA")
```

Figure 9 SPARQL query for music groups whose members are also members of ABBA. In the first triple pattern of the WHERE part, the URI of ABBA is the subject.

between the statements that are true when one statement logically follows from one or more statements. Entailment regimes specify an entailment relation between well-formed RDF graphs, assuming that a graph G entails another graph E (denoted $G \models E$) if there is a logical consequence from G to E. A regime extends the query of explicitly existing facts with facts that can be inferred using RDFS and OWL constructs (cf. Section 6), such as the extension of facts about subclasses using `rdfs:subClassOf`.

Depending on the feature set of the respective SPARQL triple store, different (or even no) entailment regimes are supported. They differ in terms of their power in the supported inference capabilities over RDFS/OWL classes and relationships. SPARQL query engines such as GraphDB adopt a materialized approach, wherein they compute the closure of the input RDF graph G over a set of relevant entailment rules R. Conversely, approaches grounded in query rewriting expand the SPARQL query itself rather than altering the RDF graph. Sub-queries are aligned with the entailment rules through backward chaining, and when the consequent of an entailment rule is matched, the antecedent of the rule is added to the query in the form of disjunctions.

Although both approaches yield equivalent answers for a given SPARQL query, their performance can diverge significantly. Materialized RDF query processing may outperform on-the-fly execution of the rewritten query, but it may consume more memory [86]. Nevertheless, various optimization techniques have been proposed to mitigate the overhead caused by the on-the-fly evaluation of entailment regimes [202]. These optimizations are required particularly in the presence of the `owl:sameAs`. This predicate corresponds to logical equivalence and involves the application of the Leibniz Inference Rule [91] to deduce all the equivalent triples entailed by equivalent resources based on `owl:sameAs` relation. This process may lead to many intermediate results, impacting the query engine's performance. Xiao et al. [221] propose query rewriting techniques to efficiently evaluate SPARQL queries with `owl:sameAs` employing equivalent SQL queries.

7.3 Federated Query Processing

Federations provide another perspective on querying linked data over multiple sources. A *federation of knowledge graphs* shares common entities while potentially providing different perspectives on those entities. Each knowledge graph within the federation operates autonomously and can be accessed through various Web interfaces, such as SPARQL endpoints or Linked Data Fragments (LDFs) [209]. SPARQL endpoints offer users the ability to execute any SPARQL query against multiple SPARQL endpoints. In contrast, LDFs enable access to specific graph patterns, such as triple patterns [208] or star-shaped graph patterns [6], allowing retrieval of fragments from an RDF knowledge graph. A *star-shaped subquery* is a conjunction of triple patterns in a SPARQL query that share the same subject variable [211]. An LDF client can submit requests to a server, which then delivers results based on a data shipping policy and partitions results into batches of specified page sizes. Query processing in a federation of graphs differs from querying a single source because it enables real-time data integration of graphs from multiple sources. For example, Figure 10 depicts a SPARQL query whose execution requires the evaluation of subqueries over three knowledge graphs: a Cancer Knowledge Graph (CKG) [7], DBpedia, and Wikidata. This query could not be executed over a single data source unless the three knowledge graphs were physically materialized into one. Subqueries with a specific shape (e. g., star-shaped subqueries) need to be identified and posed against the knowledge graph(s) that is able to answer a particular part of the query. The federated query engine has to decompose input queries into these subqueries, find a plan to execute them and collect and merge the answers from the subqueries to produce a federated answer.

A federated query engine is a system designed to execute queries over a federation of graphs. These engines leverage advanced query optimization methods to effectively decompose an input query into subqueries that can be executed over one or more graphs within the federation. Furthermore, they identify the most appropriate graphs for executing each subquery. In addition to query execution, these engines also play a crucial role in discovering and managing efficient plans that minimize the data merging costs associated with collecting information from the selected knowledge graphs.

A federated SPARQL query engine typically follows a mediator and wrapper architecture, which has been established in previous research [216, 225]. Wrappers play a crucial role in translating SPARQL subqueries into requests sent to SPARQL endpoints, while also converting the endpoint responses into internal structures that the query engine can process. The mediator, on the other hand, is responsible for rewriting the original queries into subqueries that can be executed by the data sources within the federation. Additionally, the mediator collects and merges the results obtained from evaluating the subqueries to produce the final answer to the federated query. Essentially, the mediator consists of three main components:

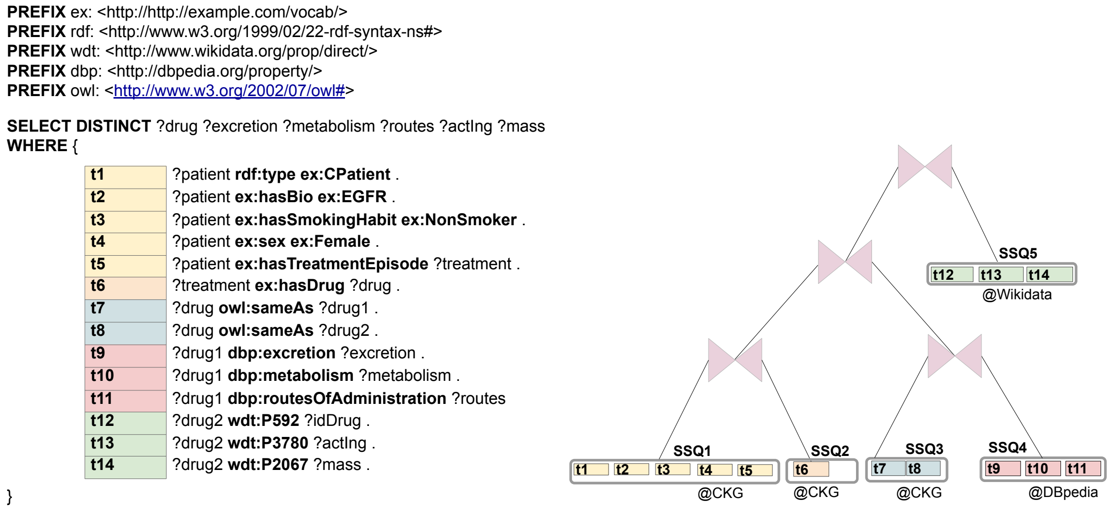
- Source selection and query decomposition. This component decomposes queries into subqueries and selects the appropriate graphs (sources) capable of executing each subquery. Simple subqueries typically consist of a list of triple patterns that can be evaluated against at least one graph. Formally, source selection corresponds to the problem of finding the minimal number of knowledge graphs from the federation that can produce a complete answer to the input query. On the other hand, query decomposition requires partitioning the triple patterns of a query into a minimal number of subqueries, such that each subquery can be executed over at least one of the selected knowledge graphs. Commonly, federated query engines follow heuristic-based methods to solve these two problems. For example, for query decomposition, heuristics based on exclusive groups [188] or star-shaped subqueries [211, 210, 148] enable to efficiently solve source selection and query decomposition in queries free of general predicates (e. g., `owl:sameAs` or `rdf:type`).

  Moreover, more general approaches (e. g., Endris et al. [63]) resort to metadata describing the star-shaped patterns existing in a knowledge graph to perform query decomposition and source selection accurately. An *exclusive group* corresponds to a set of triple patterns (corresponding to a conjunction of the triples) in a SPARQL query that can be exclusively executed over one data source, as in FedX [188]. Such a technique is implemented in the SPARQL engine FedX [188], which also optimizes the execution of *star-shaped subqueries* by identifying groups of conjunctive triple patterns that a data source can execute independently from others.

- Query optimizer. This component identifies execution plans by combining star-shaped subqueries (SSQs) and utilizing physical operators implemented by the query engine. Formally, optimizing a query corresponds to the problem of finding a physical plan for the query that minimizes the values of a utility function (e. g., execution time or memory consumption). To maximize the utility function, query optimizers consider plans with different orders of executing operators, alternative implementations of operators, such as joins, as well as particular execution alternatives for certain query types, e. g., queries involving aggregation [117]. In general, finding an optimal solution is computationally intractable [116], while the problems of constructing a *bushy tree plan* [185] and finding an optimal query decomposition over the graphs [210] are NP-Hard. A bushy tree plan is a query execution plan that represents a query as a tree structure with multiple branches or subqueries, which can also be bushy-tree plans. Query plans can be generated following the traditional optimize-then-execute paradigm or re-optimize and adapt a plan on the fly according to the conditions and availability of selected graphs [64]. Alternatively, the query optimizer may resort to a cost model to guide the search on the space of query plans and identify the one that minimizes the values of the utility function [148].
- *Query engine.* This component of a federated query engine implements the physical operators necessary to combine tuples obtained from the graphs. These physical operators are designed to support logical SPARQL operations such as JOIN, UNION, or OPTIONAL [163]. Physical operators can be empowered to adapt execution schedulers to the current conditions of a group of selected graphs. Thus, adaptivity can be achieved at the intra-operator level, where the operators can detect when graphs become blocked or data traffic bursts. Additionally, intra-operator opportunistically produce results as quickly as data arrives from the graphs, and can produce results incrementally. Some opportunistic approaches [5, 111, 78, 4] combine producing results quickly in an incremental fashion with greedy source selection so that the system stops querying additional graphs once the user's wishes, e. g., in terms of the minimum number of obtained results, are fulfilled. The physical operators implemented by the SPARQL federated query engine ANAPSID [5] implement intra-operator adaptivity, enabling results generation even when the graphs became blocked. On the other hand, inter-operator adaptive strategies can produce an answer as soon as it is computed and can keep producing intermediate results even when data from a source becomes blocked. Acosta and Vidal [4] propose adaptive query processing techniques that enable the re-ordering of a query execution plan to adjust execution schedulers to unexpected environmental conditions (e. g., changes at rates at which tuples arrive from graphs or a graph's availability).

During the query optimization process, a plan is generated as a bushy tree that comprises four join operators. This is shown in Figure 10.

The problem of query execution is defined in terms of data shipping strategies that are able to distribute query load between a server-client architecture of Linked Data Fragments. State of the art approaches tackle this issue by employing graph partitioning techniques to generate graph pattern fragments. These fragments aim to improve query performance, scalability, and workload balance. SAGE [144], smart-KG [14], brTPF [102], SkyTPF [126], SPF [6], TPF [209], and WiseKG [13] are exemplary approaches for accessing graph patterns of varying shapes. These interfaces for Linked Data Fragments may significantly accelerate query execution, reduce workload, and minimize data transfer. Moreover, query rewriting based on these fragments correctly produces all the answers of the rewritten query, i. e., the methods are sound and complete. However, it is important to note that there exists a trade-off between memory consumption and execution time improvement so that storage requirements (e. g., smart-KG) can be twice as high in comparison



```
PREFIX ex: <http://http://example.com/vocab/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX wdt: <http://www.wikidata.org/prop/direct/>
PREFIX dbp: <http://dbpedia.org/property/>
PREFIX owl: <http://www.w3.org/2002/07/owl#>

SELECT DISTINCT ?drug ?excretion ?metabolism ?routes ?actlng ?mass
WHERE {
        t1   ?patient rdf:type ex:CPatient .
        t2   ?patient ex:hasBio ex:EGFR .
        t3   ?patient ex:hasSmokingHabit ex:NonSmoker .
        t4   ?patient ex:sex ex:Female .
        t5   ?patient ex:hasTreatmentEpisode ?treatment .
        t6   ?treatment ex:hasDrug ?drug .
        t7   ?drug owl:sameAs ?drug1 .
        t8   ?drug owl:sameAs ?drug2 .
        t9   ?drug1 dbp:excretion ?excretion .
        t10  ?drug1 dbp:metabolism ?metabolism .
        t11  ?drug1 dbp:routesOfAdministration ?routes
        t12  ?drug2 wdt:P592 ?idDrug .
        t13  ?drug2 wdt:P3780 ?actlng .
        t14  ?drug2 wdt:P2067 ?mass .
}
```

(a) Federated Query **(b)** Bushy-Tree Plan

Figure 10 Federated query. a) SPARQL query comprising 14 triple patters to be executed over a federation including Cancer Knowledge Graph (CKG), DBpedia, and Wikidata. b) A query plan composed of five star-shaped subqueries SSQ1, SSQ2, SSQ3, SSQ4, and SSQ5 corresponding to the query decomposition. Each SSQ is executed over the graph that can answer the SSQ. The execution engine follows the query plan; the execution of four joins merges the SSQ answers and produces the federated query answer.

to other approaches (e. g., SAGE, TPF, or SPF), due to the need to store partitions and their associated metadata.

Because of the variety of available interfaces, some works have proposed methods for heterogeneous federations [105, 147, 146, 46], where the goal is to find efficient query execution plans in consideration of the constraints as well as strengths of the available interface.

8 Trustworthiness and Provenance of Graph Data

Trustworthiness of web pages and data on the web can be detected by various indicators, e. g., by certificates, by the placement of search engine results, and by links (forward and backward links) to other pages. However, on the Semantic Web, there are few ways for users to assess the trustworthiness of individual data. Rules can be utilized to define policies and business logic over the web of data, and transparently used to infer data that validate or do not validate these policies. The trustworthiness of inferred data can be assessed through its provenance, which encompasses metadata detailing how the data was acquired and verified [127].

The trustworthiness of data on the web can be inferred from the trustworthiness of other users ("Who said that?"), the temporal validity of facts ("When was a fact described?"), or in terms of uncertainty of statements ("To what degree is the statement true?"). Artz and Gil [10] summarize trustworthiness as follows: "Trust is not a new research topic in computer science, spanning areas as diverse as security and access control in computer networks, reliability in distributed systems, game theory and agent systems, and policies for decision-making under uncertainty. The concept of trust in these different communities varies in how it is represented, computed, and used." Although trustworthiness has long been considered in these areas, the provision and publication of data by many users to multiple sources on the Semantic Web introduces new and unique challenges.

One way of facilitating trust on the Semantic Web is to capture and provide the provenance of data with the PROV ontology (PROV-O)[60]. It captures information about which *Agents* cause data *Entities* to be processed by which *Activities*. Capturing such information requires the use of known tools for modeling metadata for RDF data, e. g., reification, singleton properties, named graphs, or RDF-star[61]. While some approaches use these constructs to capture provenance information for each triple individually [73], others exploit the fact that typically multiple triples share the same provenance [99] so that they can be combined into the same named graph encoding the provenance information only once for a set of triples. Delva et al. [54] introduce the notion of shape fragments, which entail the validation of a given shape through the neighborhood of a node, along with the node's provenance and the rationale behind its validation.

Furthermore, trustworthiness also plays a role in inference services on the Semantic Web, as data inference must consider specifications related to trustworthiness and data must be evaluated for trustworthiness. Important aspects for trustworthiness of data include [10]: the origin of the data, trust already gained based on previous interactions, ratings assigned by policies of a system, and access controls and, in some cases, security and importance of information. These aspects are realized in different systems.

In general, data provenance and trustworthiness of data on the Semantic Web have been addressed for RDF data [58, 71] as well as for OWL and rules in [58]. In addition, there are some recent approaches on supporting how-provenance for SPARQL queries [83, 106, 72] with the goal of providing users with explanations on how the answers to their queries were derived from the underlying graphs. Other work deals with access controls over distributed data on the Semantic Web [80]. Furthermore, there are approaches to computing trust values [194] and informativeness of subgraphs [123]. There are also digital signatures for graphs [19]. Analogous to digital signatures for documents, entire graphs or selected vertices and edges of a graph are provided with a digital signature to ensure the authenticity of the data and thus detect unauthorized modifications [124]. In the approach for digital graph signatures developed by Kasten et al., graph data on the Web is supported in RDF format as well as in OWL [125]. The digital graph signature is itself represented as a graph again and can thus be published together with the data on the Web. The link between the signature graph and the signed graph is established by the named graph mechanism [125], although other mechanisms are also possible. Through this mechanism, it is possible to combine and nest signed graphs. It is thus possible to re-sign already signed graphs together with other, new graph data, etc. This makes it possible to build complex chains of trust between publishers of graph data and to be able to prove the origin of data [125, 124].

9 Machine Learning on Knowledge Graphs

Machine learning on graphs is one of the most active fields of research in AI [98]. We will first consider graph embeddings for shallow representations of nodes, edges, and graphs. Then we turn our attention to neural networks for graphs (GNNs), which can compute more complex representations. Finally, we discuss developments in language models and their relationship to and integration with knowledge graphs.

[60] https://www.w3.org/TR/prov-o/

[61] https://w3c.github.io/rdf-star/

9.1 Graph Embeddings

Graph embeddings are used to compute representations of nodes, edges, and graphs for use in machine learning tasks, such as input for GNNs. The basic idea behind embeddings is to use unsupervised learning to determine a representation of nodes, for example, via their neighborhood relationships in a high-dimensional space.

9.1.1 Classical Graph Embeddings

Classic approaches to the unsupervised calculation of graph embeddings include Node2Vec [93], which uses random graph traversal to calculate how similar the start node is to nodes in its neighborhood. However, graph embeddings also include self-supervised aspects, such as in Deep Walk [164], a generalization of Node2Vec using a weighted breadth-first versus depth-first search. In Deep Walk, random traversals in the graph are also calculated starting from a node. The embeddings are then calculated using, among other things, the skip-gram idea from the embedding model Word2Vec [143] for texts. The latter provides for an element to be removed from a sequence in a self-supervised approach and predicted by a neural network. In the context of Deep Walk, the sequences are the random traversals in the graph and the elements are the nodes in the graph. An overview of classical graph embeddings can be found in Hamilton [98] and Murphy [152], among others.

9.1.2 Embeddings for Knowledge Graphs

Specific embedding models for knowledge graphs include RDF2Vec [175] and a variety of approaches based on it. In general, embeddings for knowledge graphs can be classified based on Biswas et al. [24] into translation-based models based on a distance function between nodes, models that model probabilities and uncertainty, and approaches based on semantic similarity determined by tensor decomposition.

Examples include TransE [31] as a translation-based model for representing nodes and edges in a common embedding space. TransR [134] extends TransE to support 1:N and other relationships. It uses separate embedding spaces, i. e., a separate matrix for each relation [24], and therefore requires more memory. While TransR is limited to the immediate neighborhood, PTransR extends this to include the encoding of paths. TransG [222] is based on a mixed model of Gaussian distributions to map different meanings of a relation such as `hasPart`.

While these embedding models operate on knowledge graphs, they do not map the specific semantics of Semantic Web standards such as RDFS and OWL. One approach in this direction is RotateE [197], which uses rotational transformations to map complex relationships in knowledge graphs [24], such as symmetric and antisymmetric relations, inversion, and composition. Graph embedding models that can map parts of the semantics of OWL ontologies are TransOWL [50] or OWL2Vec [44]. These map the semantics of description logic expressions such as the intersection of two classes, subclasses, etc. directly in the training of the embeddings using special loss functions. TransOWL extends previous embedding models by encoding not only the directly observed graph, but also knowledge derived from conclusions based on available axioms. To do this, TransOWL integrates the OWL axioms directly into the loss function, effectively performing inferences during training. TransOWL is derived from TransE, while its variant TransROWL is based on TransR. In addition, there is also a version that extends TransOWL with equivalence axioms and inverse relations. Another model for OWL is OWL2Vec*, which uses random walks and word embeddings to encode the semantics of OWL ontologies.

9.1.3 Discussion of Graph Embeddings

Graph embeddings have a number of disadvantages [98]. First, the calculations are slow due to high redundancy. Second, the embeddings of a graph cannot be easily transferred to new nodes or even new graphs without further learning. Therefore, graph embeddings are inherently transductive, meaning they assume that the entire graph is available at training time and can be used for unsupervised (Node2Vec) or self-supervised learning (Deep Walk). Therefore, the embeddings that have already been calculated cannot be transferred to new nodes. The inefficiency caused by redundant calculations stems from the fact that a separate embedding must be determined for each node via random traversals. In doing so, the same nodes are often visited multiple times for different embeddings. It should also be mentioned that the primary goal of graph embeddings is to determine a suitable representation of graphs when dataset-specific features are not available or are to be enriched. Therefore, embeddings do not usually have a specific downstream task such as node classification.

9.2 Neural Networks for Graphs

Graph Neural Networks (GNNs) extend the concept of graph embeddings by enabling more complex representations to be calculated based on the node features. In contrast to graph embeddings, GNNs are typically characterized by supervised or semi-supervised learning. The origins of GNNs can be traced back at least to the work of Gori et al. [89] and Scarselli et al. [181].

9.2.1 Categories of Graph Neural Networks (GNNs)

GNNs can be classified as isotropic and anisotropic, as well as transductive and inductive [76]. The majority of GNNs are based on the principle of message passing, in which features of neighboring nodes are used to compute representations of the nodes. In isotropic GNNs, all edges have the same influence on the calculation of node representations, while in anisotropic models, the edge weights and thus the representations of neighboring nodes are different or are learned.

As described above, transductive GNNs assume that the unlabeled nodes from the test set are also available at training time. These can be used in various calculations, for example for message passing, but have no influence on the error function of the model and are therefore not taken into account in the learning process using backpropagation. Examples of GNNs are GCN (isotropic and transductive), which is motivated by spectral analysis in graph theory [128]. GAT (anisotropic and inductive) learns the edge weights using attention [207]. GraphSAGE (isotropic and inductive) uses a sampler, among other things, and thus scales for large graphs. When the hyperparameters are optimized, these models are powerful representatives of GNNs with very good performance [190, 139].

In classical GNNs such as GCN, message passing is used to efficiently calculate embeddings for all nodes in the graph simultaneously [98]. As mentioned above, GNNs are typically computed through supervised or semi-supervised learning and involve a downstream task such as node or graph classification, graph regression, and edge prediction (further applications are described in Murphy [152]). Learning using GNNs is often referred to as semi-supervised because in many datasets and methods, only a few nodes per class are labeled with a gold standard label [98]; for example, less than 5% of the nodes have a label when training a GCN [128].

9.2.2 Oversquashing and Oversmoothing in GNNs

One problem with classical GNNs is that long paths cannot be mapped, leading to oversquashing and oversmoothing [98]. Oversquashing refers to when the representations of distant nodes are "squashed" together over only a few edges, leading to a loss of important information for calculating the representation of the target node. Oversmoothing refers to the incremental alignment of all representations in the GNN when, after many message-passing steps, the representations of the nodes lose their individual features. These problems can be effectively addressed by inserting additional learnable weights between the GNN layers, as in the GCNII model [45]. Other approaches, such as Graph-MLP, no longer depend on message passing and apply contrastive learning [114].

9.2.3 Homophilic and Heterophilic Graphs

Another discussion surrounding GNNs concerns the distinction between homophilic and heterophilic graphs, i. e., whether neighboring nodes are highly likely to belong to the same class or to different classes [140]. While models such as GCN assume that homophily is important for good performance, later work concludes that graphs can also be heterophilic as long as the neighborhood distribution of classes is consistent [140]. A good overview of the most important GNNs can be found in Hamilton [98] and Murphy [152], among others.

9.2.4 GNNs for Knowledge Graphs

Specific neural networks for knowledge graphs include R-GCN [186], an extension of GCN for learning relation-specific adjacency matrices. Since the number of relations in knowledge graphs can be very large [90], R-GCN approximates the relation-specific adjacency matrices using a linear combination of fewer base matrices. KGAT is a graph attention neural network based on a hybrid combination of user-item graphs and knowledge graphs [213]. The goal is to model higher-order relationships such as user-book-author-user and user-book-genre-user, and it is used for recommendation tasks.

A detailed overview of neural networks for knowledge graphs can be found, e. g., in Kumar et al. [223]. A discussion of future topics can be found in d'Amato et al. [49].

10 Language Models and Knowledge Graphs

Language models based on neural networks, in particular the Transformer architecture [206], are often distinguished into encoder models (e. g., BERT [56]), encoder-decoder models (such as T5 [172]), and decoder models (e. g., Llama [60] and GPT [160]). These differ in their capabilities and primary uses. Encoder models are particularly suitable for classification [74], while decoder models dominate the field of generative AI and chatbots [75]. However, all models share characteristics that are summarized under the term *foundation models* [30]. These include pre-training using self-supervised learning, the use of a broad corpus as training data, and the associated applicability or adaptability to a variety of downstream tasks. The size of the language models ranges from several hundred million parameters (BERT) to billions of parameters (Llama, GPT). These are therefore also referred to as small language models versus large language models.

10.1 Graph Transformer, Language Models, and Retrieval Augmented Generation

Adaptations of the transformer architecture to graphs, such as graph transformer networks [224], have already shown strong performance on downstream tasks in 2021. However, initial approaches were not yet based on pre-training models and were therefore not transferable to different downstream tasks. One reason for this is that the features between the datasets are too specific and also have different dimensionalities. Some early approaches to integrating knowledge graphs with language models have therefore aimed to encode the facts from knowledge graphs directly into the parameters of language models. Interesting approaches in this direction include SKILL [145], in which the language model T5 was trained on a textual representation of factual knowledge from knowledge graphs. SKILL shows that pre-training T5 on the Wikidata knowledge graph [212] leads to better downstream performance. The knowledge graph "disappears" into the model parameters. Changing the knowledge graph or deleting facts therefore becomes very laborous. ATLAS [119] is a retrieval-augmented method (RAG) for integrating knowledge-intensive tasks into a language model with few training examples. The goal of ATLAS is to learn facts and use them, e. g., for fact checking or asking trivia questions. An overview of methods for editing knowledge graphs in language models can be found, e. g., in Cao et al. [37].

Due to the text-based nature of language models such as T5, it was necessary for the graphs to be represented in text form. Features such as TF-IDF in GCN datasets cannot be well represented in so-called *text-attributed graphs*. Knowledge graphs in RDFS or OWL, on the other hand, are well suited due to their descriptive properties of nodes and edges. A collection of text-based graphs can be found in TAGLAS [68], among others.

10.2 Graph Foundation Models

Under the term *graph foundation models*, there are a number of approaches for graphs that meet the above requirements. The models differ in terms of their architecture, i. e., whether they are based on an existing language model, a GNN, or a combination of language model and GNN [136]. The Graph Language Model (GLM) [165], which is based on Google's T5 language model, belongs to the first category. The attention mechanism of T5's transformer layer has been modified so that masking takes into account the adjacency of nodes. This modification of the attention mask also distinguishes GLM from the above-mentioned approaches such as SKILL. For T5, the input graph is converted into a Levi graph and then processed as text, like any other prompt, in the T5 model. Due to the nature of GLM, it is suitable for processing both text and graphs natively, but is limited to text-attributed graphs.

Models such as PRODIGY, PSP, and GraphAny pursue an approach based on GNNs. The first two, PRODIGY [115] and PSP [82], can be understood as native graph-based foundation models that map a graph prompt in analogy to a text prompt. Based on a pre-trained model, similar to language models, several graphs can be given as examples (*context learning*) along with the task to be solved. Unlike PRODIGY, PSP works with prototypes that represent the graphs. GraphAny [227] uses a combination of different linear GNNs based on the Simplified GCN Model (SGC) [219]. In order to meet the requirement of being usable for various downstream tasks, GraphAny has a mechanism for handling different dimensionalities of node features in graph datasets. GraphAny is very efficient due to the use of linear models. In addition, training on a few examples is sufficient to achieve strong performance on other datasets with different features. Approaches in the field of combining language models and GNNs include [220, 135, 130].

An overview of graph foundation models is provided by Liu et al. [137] and Fan et al. [67].

11 Applications

With the increasing spread and use of semantic and linked data on the Web, the requirements for Semantic Web applications have increased at the same time as their application possibilities. The general requirements for applications based on semantic data on the Web are given by their flexible and diverse representation and descriptions. Applications that use data from relational databases or XML documents can start from a fixed schema. However, this cannot be assumed for data on the Web. Often, neither the data sources nor the type and amount of data in a source are fully known. The dynamics of semantic data on the Web must be taken into account by applications accordingly, both when querying and aggregating data, and when visualizing data. Thus, the real challenge of Semantic Web applications is to guarantee the best possible flexibility of the application to take into account the dynamics of data sources, data, and schemas during input, processing, and output.

In the following, selected examples of Semantic Web applications or application areas are presented. They illustrate how flexibility and quality of search, integration, aggregation, and presentation of data from the Web can be implemented. At the same time, they show the potential of Semantic Web applications. First, uniform vocabularies and schemas are presented using the example of *schema.org*. These serve as a basis for semantic search to provide search engines with information about the meaning of web document content. The search and integration of data from different sources is supported by *Sig.ma*, a semantic web browser. Other applications provide semantic search through other representation formalisms, e. g., *Knowledge Graphs*). Subsequently, the *Facebook Graph-API*, an application programming interface (*API*) to the Facebook (Knowledge) Graph, is introduced.

11.1 Vocabularies and Schemas: Schema.org

In HTML documents, the structure and composition of pages can be described with tags, but not the meaning of the information. Vocabulary, schemas, and microdata can be used as mark-up in HTML documents to describe information about page content and its meaning in a way that search engines can process this information.

Schema.org[62] is a collection of vocabularies and schemas to enrich HTML pages with additional information. The vocabulary of *Schema.org* includes a set of classes and their properties. A universal class “thing” is the most general, which is a kind of umbrella term for all classes. Other common classes are *Organization*, *Person*, *Event*, and *Place*. Properties are used to describe classes in more detail. For example, a person has the properties such as name, address, and date of birth.

In addition to vocabularies, Schema.org also specifies the use of HTML microdata, with the goal of representing data in HTML documents in as unambiguous a form as possible so that search engines can interpret it correctly. An example of this is formats for unique dates and times, which can also describe intervals to indicate the duration of events.

Schema.org is supported by the search engines Bing, Google, and Yandex, among others. There are extensions and libraries for various programming languages, including PHP, JavaScript, Ruby, and Python, to create web pages and web applications using vocabularies and microdata from Schema.org. Likewise, there are mappings from Schema.org vocabularies and microdata to RDFS.

11.2 Semantic Search

A classic web browser enables the display of web pages. A semantic web browser goes one step further by additionally allowing the user to visualize the underlying information of individual pages, for example in the form of RDF metadata. Semantic Web browsers are also referred to as hyperdata browsers because they allow navigation between data while also allowing one to explore the connection to information about that data. Thus, ordinary users can use and exploit Semantic Web data for their information search.

[62] `http://schema.org`

Sig.ma [203] was an application for (browsing) Semantic Web data, which may come from multiple distributed data sources. Sig.ma provided an API for automatically integrating multiple data sources on the Web. The requested data sources describe information in RDF. A search in Sig.ma was initiated by a textual query from the user. Entities such as people, places, or products can be searched for. Results of a query are presented in aggregated form, that is, properties of the searched entity, such as a person, are presented in aggregated form from different data sources. For example, in a person search, information such as e-mail address, address, or current employer can be displayed. In addition to the actual information, links to the underlying data sources are also displayed to allow users to navigate to refine their search. Sig.ma also supported structured queries in which specific characteristics can be requested for an entity, such as contact information for a specific person.

Queries to data sources occur in parallel. The results from each data source in the form of RDF graphs are summarized by using properties of links in RDF data, such as `owl:sameAs`, or inverse-functional predicates. When searching data sources, techniques such as indexes, logical inference, and heuristics are used for data aggregation. OntoBroker[63] [53] and OntoEdit [198] are ontology editors with search and inference systems for ontologies. Using OntoBroker, complex queries over distributed Semantic Web resources, e. g., represented in OWL, RDF, RDFS, SPARQL, and also F-Logic) can be efficiently processed.

11.3 Knowledge Graphs and Wikidata

There is an increasing number of knowledge bases and representations of structured data. For example, the secondary database Wikidata[64] [212]. A secondary database includes, in addition to the (actual) statements, relationships to their sources and other databases (called secondary information). Wikidata is a shared database between Wikipedia and Wikimedia. Wikidata mainly contains a collection of objects, which are represented as triples over the objects' properties and the corresponding values. Semantic MediaWiki[65] is an extension of MediaWiki. It serves as a flexible knowledge base and knowledge management system. Semantic MediaWiki extends a classic wiki with the ability to enrich content in a machine-readable way using semantic annotations.

Another knowledge base was Freebase [29, 28], also an open and collaborative platform initiated in 2007 and acquired by Google in 2010. The content from Freebase was taken from various sources, including parts from the MusicBrainz ontology mentioned earlier. The success and widespread use of Wikidata prompted Google to migrate Freebase to Wikidata [199]. This strengthened the goal to develop a comprehensive, collaborative basis of structured data.

Google offers a semantic search function with Google Knowledge Graph[66,67]. A knowledge graph, like an RDF graphs, is a set of triples representing links between entities. This forms a semantic database. Possible entity types are described on schema.org, among others. If a search term occurs in a query, the corresponding entity is searched for in the knowledge graph. Starting from this entity, it is then possible to navigate to further entities by means of the links.

[63] https://www.semafora-systems.com/ontobroker-and-ontostudio-x
[64] https://www.wikidata.org/wiki/Wikidata:Introduction/de
[65] https://www.semantic-mediawiki.org/wiki/Semantic_MediaWiki
[66] http://www.google.com/insidesearch/features/search/knowledge.html
[67] https://developers.google.com/knowledge-graph/

11.4 API-Access to Social Networks

A social network is essentially a graph in which connections are formed from users to other users, e. g., in the form of a friendship relationship or to events and groups. Facebook's Graph API describes a programming interface to the Facebook Graph (called *Open Graph*). Within the graph, people, events, pages, and photos are represented as objects, with each object having a unique identifier. For example, `https://graph.facebook.com/abba` is the identifier of ABBA's Facebook page. There are also unique identifiers for the possible relationship types of an object, which allow navigating from one object to all connected objects with respect to a particular relationship.

The Graph API allows one to navigate the Facebook Graph and read objects, including their properties and relationships to other objects, as well as creating new objects in the Facebook Graph and deploying applications. The API also supports requests for an object's metadata, such as *when* and *by whom* an object was created.

12 Impact for Practitioners

Linking and using graph data on the Web has become a widespread practice. Today, there is a large amount of open data in various formats and domains, such as bibliographic information management, bioinformatics, and e-government. DBpedia is the central hub in this context, around which different datasets and domains are grouped (cf. [25]). This is illustrated, e. g., by the tremendous growth of the Linked Open Data Cloud[68] since 2007. Two of the latest notable supporters of graph-based data are online auctioneer eBay with their graph database[69] and the U.S. space agency NASA with the unification of internal distributed case databases as knowledge graphs[70]. These and other success stories of the Semantic Web in industries and industry-scale knowledge graphs are described by Noy et al. [154]. Further analyses and surveys arguing about the importance but also challenges of using graph data can be found in the literature like the 2020 survey of Sahu et al. [179] and the 2021 reflection about the future of graphs by Sakr et al. [180]. The usefulness of knowledge graphs and semantic-based data modeling for complex systems is also discussed in the 2024 book by Abonyi et al. [2]. The importance of graph databases is also reflected by the Forbes business magazine, which predicted in 2019 that graph databases would be the next mainstream database technology[71].

Regarding lightweight open graph data, Schema.org defines schemas for modeling data on web pages to provide information about the underlying data structures and meaning of the data. Search engines can use this additional information to better analyze the content of web pages. As mentioned above, Schema.org is supported by search engines such as Bing, Google, and Yandex. Studies on selected sources have shown that web pages among the top 10 results have up to 15 % higher click-through rate[72]. Other companies like BestBuy.com even report up to 30 % higher click-through rates since adding semantic data to their websites (cf. Section 11) in 2009. BestBuy.com uses the GoodRelations vocabulary[73] to describe online offers. Similarly, Google

[68] The growth of the Linked Open Data Cloud is documented at: `http://linkeddata.org/`.
[69] `https://github.com/eBay/akutan`
[70] `https://blog.nuclino.com/why-nasa-converted-its-lessons-learned-database-into-a-knowledge-graph`
[71] `https://www.forbes.com/sites/cognitiveworld/2019/07/18/graph-databases-go-mainstream`
[72] `http://developer.yahoo.net/blog/archives/2008/07/`
[73] `http://www.heppnetz.de/projects/goodrelations/`

uses semantic data from online commerce portals that use the GoodRelations vocabulary and takes it into account when searching[74].

Another success is the publication of government data. For example, the U.S. government makes government data publicly available with data.gov[75], and U.S. Census[76] publishes statistical data about the United States. In the UK, data.gov.uk[77] is a key part of a program to increase data transparency in the public sector. The European Commission operates data.europa.eu[78], a European data portal with metadata about the member states. Among others, it provides a SPARQL endpoint to access the data.

Finally, a strong growth of semantic biomedical data on the Web can be noted. As part of Bio2RDF[79], many bioinformatics databases have been linked. Transinsight GmbH offers the knowledge-based search engine GoPubMed[80] to find biomedical research articles. Ontologies are used for searching.

Regarding more heavyweight ontologies in OWL, there has also been movement in recent years. In addition to numerous research-derived inference engines such as Pellet and Hermit mentioned above, inference mechanisms for OWL can now be found in commercial graph databases such as neo4j[81]. Furthermore, pattern-based core ontologies can also be found in software development workflows [187]. The development and use of core ontologies is part of a continuous delivery process that is used in practice.

13 Summary and Outlook

The Semantic Web consists of a variety of techniques that have been heavily influenced by long-term artificial intelligence research and its results. The current state is also driven by an industry uptake under the umbrella term of Knowledge Graphs and reflected in various activities as described. In summary, therefore, it can be observed that semantic data on the Web is having a real impact on commercial providers of products and services, as well as on governments and public administrations.

Despite all the research and industrial developments, the full potential of the Semantic Web has not yet been exploited. Some important components of the Semantic Web architecture are still being explored, such as data provenance and trustworthiness. Below, we describe three example directions for future work.

- Neuro-symbolic systems: As mentioned in the introduction, we see as an important direction of future work the combination of symbolic AI and subsymbolic AI. By combining the strength of Large Language Models (LLM), i. e., generative AI, in processing and generating natural language text and accessing structured data and logical reasoning capabilities of the Semantic Web, a next step towards the vision of automated agents that perform complex planning tasks may be reached. An example is performing A* search with an LLM [228]. Specifically, LLMs might comprehensively capture and acquire human knowledge [55], but current LLMs lack responding to simple questions of non-existing facts in their training data [55], may not contain all facts [196], and thus return less accurate answers [113]. To leverage the distinct capabilities

[74] http://www.ebusiness-unibw.org/wiki/GoodRelationsInGoogle
[75] http://www.data.gov/
[76] http://www.rdfabout.com/demo/census/
[77] http://data.gov.uk
[78] https://data.europa.eu/
[79] http://bio2rdf.org/
[80] http://www.gopubmed.org/
[81] https://neo4j.com/blog/neo4j-rdf-graph-database-reasoning-engine/

of both LLMs and the Semantic Web, the integration of neuro-symbolic systems appears to offer a viable solution [161]. Neuro-symbolic systems could also address the problem that LLMs' output is based on the most probable answer, which sometimes leads to wrong answers – often referred to as "hallucinations" [17, 110, 196].

- Natural interfaces between machine and users: A key to successful applications of the Semantic Web is intuitive user interfaces. Users must be offered applications that are intuitive and easy to use. This includes improving interfaces based on natural language for formulating queries and accessing structured data stored in SPARQL endpoints. Again, the use and deeper integration of LLMs with Knowledge Graphs shows a promising direction.
- Semantic Web components: There are still components of the architecture (see Section 3) where active development and research are conducted. Most notably, there are crypto and trust. Recent new W3C standards such as DID and Verifiable Credentials have been developed. However, one can expect more work and development in this direction.

Finally, we like to point to existing literature discussing the future directions of research on the Semantic Web [39] and Knowledge Graphs [59]. Breit et al. [33] conducted a survey on the fusion of Semantic Web and Machine Learning, exploring the opportunities arising from the convergence of these two paradigms.

References

1 OWL 2 Web Ontology Language Document Overview . W3C recommendation, W3C, October 2009. http://www.w3.org/TR/owl2-overview/.

2 János Abonyi, László Nagy, and Tamás Ruppert, editors. *Ontology-Based Development of Industry 4.0 and 5.0 Solutions for Smart Manufacturing and Production: Knowledge Graph and Semantic Based Modeling and Optimization of Complex Systems.* Springer, 2024.

3 Ghadeer Abuoda, Christian Aebeloe, Daniele Dell'Aglio, Arthur Keen, and Katja Hose. Star-Bench: Benchmarking RDF-star Triplestores. In *QuWeDa icw. ISWC 2023*, volume 3565 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

4 Maribel Acosta and Maria-Esther Vidal. Networks of linked data eddies: An adaptive web query processing engine for RDF data. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*, volume 9366 of *Lecture Notes in Computer Science*, pages 111–127. Springer, 2015.

5 Maribel Acosta, Maria-Esther Vidal, Tomas Lampo, Julio Castillo, and Edna Ruckhaus. ANAPSID: an adaptive query processing engine for SPARQL endpoints. 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 18–34. Springer, 2011.

6 Christian Aebeloe, Ilkcan Keles, Gabriela Montoya, and Katja Hose. Star Pattern Fragments: Accessing Knowledge Graphs through Star Patterns. abs/2002.09172, 2020. URL: https://arxiv.org/abs/2002.09172.

7 Fotis Aisopos, Samaneh Jozashoori, Emetis Niazmand, Disha Purohit, Ariam Rivas, Ahmad Sakor, Enrique Iglesias, Dimitrios Vogiatzis, Ernestina Menasalvas, Alejandro Rodríguez González, Guillermo Vigueras, Daniel Gómez-Bravo, Maria Torrente, Roberto Hernández López, Mariano Provencio Pulla, Athanasios Dalianis, Anna Triantafillou, Georgios Paliouras, and Maria-Esther Vidal. Knowledge graphs for enhancing transparency in health data ecosystems. *Semantic Web*, 14(5):943–976, 2023.

8 Medina Andreşel, Julien Corman, Magdalena Ortiz, Juan L. Reutter, Ognjen Savković, and Mantas Šimkus. Stable model semantics for recursive shacl. In *ACM–The Web Conference*, pages 1570–1580, 2020.

9 Julián Arenas-Guerrero, David Chaves-Fraga, Jhon Toledo, María S. Pérez, and Oscar Corcho. Morph-KGC: Scalable knowledge graph materialization with mapping partitions. *Semantic Web*, 2022. doi:10.3233/SW-223135.

10 Donovan Artz and Yolanda Gil. A Survey of Trust in Computer Science and the Semantic Web. *J. Web Sem.*, 5(2):58–71, 2007.

11 Dylan Van Assche, Thomas Delva, Gerald Haesendonck, Pieter Heyvaert, Ben De Meester, and Anastasia Dimou. Declarative RDF graph generation from heterogeneous (semi-)structured data: A systematic literature review. *J. Web Semant.*, 75:100753, 2023. doi:10.1016/j.websem.2022.100753.

12 S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A Nucleus for a Web of Open Data. In *Semantic Web Conference and Asian Semantic Web Conference*, pages 722–735, November 2008. URL: http://dx.doi.org/10.1007/978-3-540-76298-0_52, doi:10.1007/978-3-540-76298-0_52.

13 Amr Azzam, Christian Aebeloe, Gabriela Montoya, Ilkcan Keles, Axel Polleres, and Katja Hose. WiseKG: Balanced Access to Web Knowledge Graphs. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 1422–1434. ACM / IW3C2, 2021.

14 Amr Azzam, Javier D. Fernández, Maribel Acosta, Martin Beno, and Axel Polleres. SMART-KG: hybrid shipping for SPARQL querying on the web. In Yennun Huang, Irwin King, Tie-Yan Liu, and Maarten van Steen, editors, *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020*, pages 984–994. ACM / IW3C2, 2020.

15 Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

16 Franz Baader and Ulrike Sattler. An overview of tableau algorithms for description logics. *Stud Logica*, 69(1):5–40, 2001. doi:10.1023/A:1013882326814.

17 Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, Quyet V. Do, Yan Xu, and Pascale Fung. A Multitask, Multilingual, Multimodal Evaluation of ChatGPT on Reasoning, Hallucination, and Interactivity. *CoRR*, abs/2302.04023, 2023.

18 David Beckett, Tim Berners-Lee, Eric Prud'hommeaux, and Gavin Carothers. Terse RDF Triple Language, 2014. http://www.w3.org/TR/turtle/.

19 Luigi Bellomarini, Markus Nissl, and Emanuel Sallinger. Blockchains as knowledge graphs - blockchains for knowledge graphs (vision paper). In *Proceedings of the International Workshop on Knowledge Representation and Representation Learning co-located with the 24th European Conference on Artificial Intelligence (ECAI 2020), Virtual Event, September, 2020*, volume 3020 of *CEUR Workshop Proceedings*, pages 43–51. CEUR-WS.org, 2020.

20 T. Berners-Lee. Universal Resource Identifiers in WWW: A Unifying Syntax for the Expression of Names and Addresses of Objects on the Network as used in the World-Wide Web. RFC 1630, Internet Engineering Task Force, June 1994. URL: http://www.rfc-editor.org/rfc/rfc1630.txt.

21 Tim Berners-Lee, Roy T. Fielding, and Larry M Masinter. Uniform Resource Identifier (URI): Generic Syntax. Technical Report 3986, January 2005. URL: https://www.rfc-editor.org/info/rfc3986, doi:10.17487/RFC3986.

22 Tim Berners-Lee, James Hendler, and Orla Lassila. The Semantic Web. *Scientific American*, pages 1–4, 2001.

23 Tim Berners-Lee, Larry M Masinter, and Mark P. McCahill. Uniform Resource Locators (URL). Technical Report 1738, December 1994. URL: https://www.rfc-editor.org/info/rfc1738, doi:10.17487/RFC1738.

24 Russa Biswas, Lucie-Aimée Kaffee, Michael Cochez, Stefania Dumbrava, Theis E. Jendal, Matteo Lissandrini, Vanessa Lopez, Eneldo Loza Mencía, Heiko Paulheim, Harald Sack, Edlira Kalemi Vakaj, and Gerard de Melo. Knowledge Graph Embeddings: Open Challenges and Opportunities. *Transactions on Graph Data and Knowledge*, 1(1):4:1–4:32, 2023. URL: https://drops.dagstuhl.de/entities/document/10.4230/TGDK.1.1.4, doi:10.4230/TGDK.1.1.4.

25 Christian Bizer. The Emerging Web of Linked Data. *IEEE Intelligent Systems*, 24(5):87–92, 2009.

26 Eva Blomqvist. Ontocase-automatic ontology enrichment based on ontology design patterns. In *International Semantic Web Conference*, pages 65–80, 2009.

27 Till Blume. *Semantic structural graph summaries for evolving and distributed graphs*. PhD thesis, University of Ulm, Germany, 2022. URL: https://nbn-resolving.org/urn:nbn:de:bsz:289-oparu-46050-1.

28 Kurt D. Bollacker, Robert P. Cook, and Patrick Tufts. Freebase: A shared database of structured general human knowledge. In *Proceedings of the Twenty-Second AAAI Conference on Artificial Intelligence, July 22-26, 2007, Vancouver, British Columbia, Canada*, pages 1962–1963. AAAI Press, 2007. URL: http://www.aaai.org/Library/AAAI/2007/aaai07-355.php.

29 Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In Jason Tsong-Li Wang, editor, *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM, 2008. doi:10.1145/1376616.1376746.

30 Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ B. Altman, Simran Arora, and et al. On the opportunities and risks of foundation models. *CoRR*, abs/2108.07258, 2021. URL: https://arxiv.org/abs/2108.07258, arXiv:2108.07258.

31 Antoine Bordes, Nicolas Usunier, Alberto García-Durán, Jason Weston, and Oksana Yakhnenko. Translating embeddings for modeling multi-relational data. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 2787–2795, 2013. URL: https://proceedings.neurips.cc/paper/2013/hash/1cecc7a77928ca8133fa24680a88d2f9-Abstract.html.

32 Stefano Borgo and Claudio Masolo. *Handbook on Ontologies*, chapter Foundational choices in DOLCE. Springer, 2nd edition, 2009.

33 Anna Breit, Laura Waltersdorfer, Fajar J. Ekaputra, Marta Sabou, Andreas Ekelhart, Andreea Iana, Heiko Paulheim, Jan Portisch, Artem Revenko, Annette Ten Teije, and Frank Van Harmelen. Combining machine learning and semantic web: A systematic mapping study. *ACM Comput. Surv.*, 55(14s), jul 2023. doi:10.1145/3586163.

34 Jeen Broekstra, Arjohn Kampman, and Frank Van Harmelen. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema. In

International Semantic Web Conference, pages 54–68. Springer, 2002.

35 Pere-Lluís Huguet Cabot and Roberto Navigli. REBEL: relation extraction by end-to-end language generation. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021*, pages 2370–2381. Association for Computational Linguistics, 2021. URL: `https://doi.org/10.18653/v1/2021.findings-emnlp.204`, `doi:10.18653/V1/2021.FINDINGS-EMNLP.204`.

36 Diego Calvanese, Benjamin Cogrel, Sarah Komla-Ebri, Roman Kontchakov, Davide Lanti, Martin Rezk, Mariano Rodriguez-Muro, and Guohui Xiao. Ontop: Answering SPARQL queries over relational databases. *Semantic Web*, 8(3):471–487, 2017. `doi:10.3233/SW-160217`.

37 Boxi Cao, Hongyu Lin, Xianpei Han, and Le Sun. The life cycle of knowledge in big language models: A survey. *Mach. Intell. Res.*, 21(2):217–238, 2024. URL: `https://doi.org/10.1007/s11633-023-1416-x`, `doi:10.1007/S11633-023-1416-X`.

38 Jeremy J. Carroll, Christian Bizer, Patrick J. Hayes, and Patrick Stickler. Semantic web publishing using named graphs. In Jennifer Golbeck, Piero A. Bonatti, Wolfgang Nejdl, Daniel Olmedilla, and Marianne Winslett, editors, *Proceedings of the ISWC*04 Workshop on Trust, Security, and Reputation on the Semantic Web, Hiroshima, Japan, November 7, 2004*, volume 127 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2004. URL: `https://ceur-ws.org/Vol-127/paper2.pdf`.

39 Irene Celino and Heiko Paulheim. The time traveler's guide to semantic web research: Analyzing fictitious research themes in the ESWC "next 20 years" track. *CoRR*, abs/2309.13939, 2023. `doi:10.48550/arXiv.2309.13939`.

40 David Chaves-Fraga, Kemele M. Endris, Enrique Iglesias, Óscar Corcho, and Maria-Esther Vidal. What are the parameters that affect the construction of a knowledge graph? In *On the Move to Meaningful Internet Systems: OTM 2019 Conferences - Confederated International Conferences: CoopIS, ODBASE, C&TC 2019, Rhodes, Greece, October 21-25, 2019, Proceedings*, volume 11877 of *Lecture Notes in Computer Science*, pages 695–713. Springer, 2019. `doi:10.1007/978-3-030-33246-4\_43`.

41 David Chaves-Fraga, Edna Ruckhaus, Freddy Priyatna, Maria-Esther Vidal, and Óscar Corcho. Enhancing virtual ontology based access over tabular data with Morph-CSV. *Semantic Web*, 12(6):869–902, 2021. `doi:10.3233/SW-210432`.

42 Michelle Cheatham and Catia Pesquita. *Semantic Data Integration*, pages 263–305. Springer International Publishing, Cham, 2017. `doi:10.1007/978-3-319-49340-4_8`.

43 Artem Chebotko, Shiyong Lu, and Farshad Fotouhi. Semantics preserving SPARQL-to-SQL translation. *Data Knowl. Eng.*, 68(10):973–1000, 2009. `doi:10.1016/j.datak.2009.04.001`.

44 Jiaoyan Chen, Pan Hu, Ernesto Jiménez-Ruiz, Ole Magnus Holter, Denvar Antonyrajah, and Ian Horrocks. Owl2vec*: embedding of OWL ontologies. *Mach. Learn.*, 110(7):1813–1845, 2021. URL: `https://doi.org/10.1007/s10994-021-05997-6`, `doi:10.1007/S10994-021-05997-6`.

45 Ming Chen, Zhewei Wei, Zengfeng Huang, Bolin Ding, and Yaliang Li. Simple and deep graph convolutional networks. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 1725–1735. PMLR, 2020. URL: `http://proceedings.mlr.press/v119/chen20v.html`.

46 Sijin Cheng and Olaf Hartig. Towards Query Processing over Heterogeneous Federations of RDF Data Sources. In *ESWC (Satellite Events)*, volume 13384 of *Lecture Notes in Computer Science*, pages 57–62. Springer, 2022.

47 Julien Corman, Fernando Florenzano, Juan L. Reutter, and Ognjen Savković. Validating SHACL Constraints over a SPARQL Endpoint. In *International Semantic Web Conference ISWC*, pages 145–163. Springer, 2019.

48 Julien Corman, Juan L. Reutter, and Ognjen Savković. Semantics and Validation of recursive SHACL. In *International Semantic Web Conference ISWC*, pages 318–336. Springer, 2018.

49 Claudia d'Amato, Louis Mahon, Pierre Monnin, and Giorgos Stamou. Machine learning and knowledge graphs: Existing gaps and future research challenges. *TGDK*, 1(1):8:1–8:35, 2023. `doi:10.4230/TGDK.1.1.8`.

50 Claudia d'Amato, Nicola Flavio Quatraro, and Nicola Fanizzi. Injecting background knowledge into embedding models for predictive tasks on knowledge graphs. In Ruben Verborgh, Katja Hose, Heiko Paulheim, Pierre-Antoine Champin, Maria Maleshkova, Óscar Corcho, Petar Ristoski, and Mehwish Alam, editors, *The Semantic Web - 18th International Conference, ESWC 2021, Virtual Event, June 6-10, 2021, Proceedings*, volume 12731 of *Lecture Notes in Computer Science*, pages 441–457. Springer, 2021. `doi:10.1007/978-3-030-77385-4\_26`.

51 Souripriya Das, Seema Sundara, and Richard Cyganiak. R2RML: RDB to RDF Mapping Language, 2012. https://www.w3.org/TR/r2rml/.

52 Jeremy Debattista, Christoph Lange, Sören Auer, and Dominic Cortis. Evaluating the quality of the LOD cloud: An empirical investigation. *Semantic Web*, 9(6):859–901, 2018. `doi:10.3233/SW-180306`.

53 Stefan Decker, Michael Erdmann, Dieter Fensel, and Rudi Studer. *Database Semantics: Semantic Issues in Multimedia Systems*, chapter Ontobroker: Ontology Based Access to Distributed and Semi-Structured Information. Springer, 1999.

54 Thomas Delva, Anastasia Dimou, Maxime Jakubowski, and Jan Van den Bussche. Data provenance for SHACL. In *Proceedings 26th International Conference on Extending Database Technology, EDBT 2023, Ioannina, Greece, March 28-31, 2023*, pages 285–297. OpenProceedings.org, 2023. `doi:10.48786/edbt.2023.23`.

55 Peter J. Denning. The smallness of large language models. *Commun. ACM*, 66(9):24–27, aug 2023. `doi:10.1145/3608966`.

56 Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. URL: `https://doi.org/10.18653/v1/n19-1423`, `doi:10.18653/V1/N19-1423`.

57 Anastasia Dimou, Miel Vander Sande, Pieter Colpaert, Ruben Verborgh, Erik Mannens, and Rik Van de Walle. RML: A generic language for integrated RDF mappings of heterogeneous data. In Christian Bizer, Tom Heath, Sören Auer, and Tim Berners-Lee, editors, *Proceedings of the Workshop on Linked Data on the Web co-located with the 23rd International World Wide Web Conference (WWW 2014), Seoul, Korea, April 8, 2014*, volume 1184 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2014. URL: `https://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf`.

58 Renata Queiroz Dividino, Simon Schenk, Sergej Sizov, and Steffen Staab. Provenance, Trust, Explanations - and all that other Meta Knowledge. *KI*, 23(2):24–30, 2009.

59 Xin Luna Dong. Generations of knowledge graphs: The crazy ideas and the business impact. *CoRR*, abs/2308.14217, 2023. `doi:10.48550/arXiv.2308.14217`.

60 Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, and others. The llama 3 herd of models. *CoRR*, abs/2407.21783, 2024. URL: `https://doi.org/10.48550/arXiv.2407.21783`, `arXiv:2407.21783`, `doi:10.48550/ARXIV.2407.21783`.

61 M. Duerst and M. Suignard. Internationalized resource identifiers (IRIs). RFC 3987, Internet Engineering Task Force, January 2005. URL: `http://www.rfc-editor.org/rfc/rfc3987.txt`.

62 Marc Ehrig. *Ontology Alignment: Bridging the Semantic Gap*, volume 4 of *Semantic Web and Beyond*. Springer, 2007.

63 Kemele M. Endris, Mikhail Galkin, Ioanna Lytra, Mohamed Nadjib Mami, Maria-Esther Vidal, and Sören Auer. Querying interlinked data by bridging RDF molecule templates. *Trans. Large Scale Data Knowl. Centered Syst.*, 39:1–42, 2018.

64 Kemele M. Endris, Maria-Esther Vidal, and Damien Graux. Federated query processing. In Valentina Janev, Damien Graux, Hajira Jabeen, and Emanuel Sallinger, editors, *Knowledge Graphs and Big Data Processing*, volume 12072 of *Lecture Notes in Computer Science*, pages 73–86. Springer, 2020. `doi:10.1007/978-3-030-53199-7\_5`.

65 Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*, chapter Classifications of ontology matching techniques. Springer, 2007.

66 Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer, 2007.

67 Wenqi Fan, Shijie Wang, Jiani Huang, Zhikai Chen, Yu Song, Wenzhuo Tang, Haitao Mao, Hui Liu, Xiaorui Liu, Dawei Yin, and Qing Li. Graph machine learning in the era of large language models (llms). *CoRR*, abs/2404.14928, 2024. URL: `https://doi.org/10.48550/arXiv.2404.14928`, `arXiv:2404.14928`, `doi:10.48550/ARXIV.2404.14928`.

68 Jiarui Feng, Hao Liu, Lecheng Kong, Yixin Chen, and Muhan Zhang. TAGLAS: an atlas of text-attributed graph datasets in the era of large graph and language models. *CoRR*, abs/2406.14683, 2024. URL: `https://doi.org/10.48550/arXiv.2406.14683`, `arXiv:2406.14683`, `doi:10.48550/ARXIV.2406.14683`.

69 Sebastián Ferrada, Benjamin Bustos, and Aidan Hogan. Extending SPARQL with similarity joins. In *ISWC (1)*, volume 12506 of *Lecture Notes in Computer Science*, pages 201–217. Springer, 2020.

70 Mónica Figuera, Philipp D. Rohde, and Maria-Esther Vidal. Trav-shacl: Efficiently validating networks of SHACL constraints. In Jure Leskovec, Marko Grobelnik, Marc Najork, Jie Tang, and Leila Zia, editors, *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*, pages 3337–3348. ACM / IW3C2, 2021. `doi:10.1145/3442381.3449877`.

71 Giorgos Flouris, Irini Fundulaki, Panagiotis Pediaditis, Yannis Theoharis, and Vassilis Christophides. Coloring RDF Triples to Capture Provenance. In *International Semantic Web Conference*, volume 5823 of *LNCS*, pages 196–212. Springer, 2009.

72 Luis Galárraga, Daniel Hernández, Anas Katim, and Katja Hose. Visualizing how-provenance explanations for SPARQL queries. In *WWW (Companion Volume)*, pages 212–216. ACM, 2023.

73 Luis Galárraga, Kim Ahlstrøm Meyn Mathiassen, and Katja Hose. QBOAirbase: The European Air Quality Database as an RDF Cube. In *ISWC (Posters, Demos & Industry Tracks)*, volume 1963 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2017.

74 Lukas Galke and Ansgar Scherp. Bag-of-words vs. graph vs. sequence in text classification: Questioning the necessity of text-graphs and the surprising strength of a wide MLP. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4038–4051. Association for Computational Linguistics, 2022. URL: `https://doi.org/10.18653/v1/2022.acl-long.279`, `doi:10.18653/V1/2022.ACL-LONG.279`.

75 Lukas Galke, Ansgar Scherp, Andor Diera, Bao Xin Lin, Bhakti Khera, Tim Meuser, Tushar Singhal, and Fabian Karl. A review of methods for single-label and multi-label text classification. *CoRR*, abs/2204.03954, 2022. URL: `https://doi.org/10.48550/arXiv.2204.03954`, `arXiv:2204.03954`, `doi:10.48550/ARXIV.2204.03954`.

76 Lukas Galke, Iacopo Vagliano, Benedikt Franke, Tobias Zielke, Marcel Hoffmann, and Ansgar Scherp. Lifelong learning on evolving graphs under the constraints of imbalanced classes and new classes. *Neural Networks*, 164:156–176, 2023. URL: `https://doi.org/10.1016/j.neunet.2023.04.022`, `doi:10.1016/J.NEUNET.2023.04.022`.

77 Mikhail Galkin, Sören Auer, Maria-Esther Vidal, and Simon Scerri. Enterprise knowledge graphs: A semantic approach for knowledge management in the next generation of enterprise information systems. In Slimane Hammoudi, Michal Smialek, Olivier Camp, and Joaquim Filipe, editors, *ICEIS 2017 - Proceedings of the 19th International Conference on Enterprise Information Systems, Volume 2, Porto, Portugal, April 26-29, 2017*, pages 88–98. SciTePress, 2017. doi:10.5220/0006325200880098.

78 Mikhail Galkin, Kemele M. Endris, Maribel Acosta, Diego Collarana, Maria-Esther Vidal, and Sören Auer. Smjoin: A multi-way join operator for SPARQL queries. In Rinke Hoekstra, Catherine Faron-Zucker, Tassilo Pellegrini, and Victor de Boer, editors, *Proceedings of the 13th International Conference on Semantic Systems, SEMANTiCS 2017, Amsterdam, The Netherlands, September 11-14, 2017*, pages 104–111. ACM, 2017. doi:10.1145/3132218.3132220.

79 Aldo Gangemi and Valentina Presutti. *Handbook on Ontologies*, chapter Ontology Design Patterns. Springer, 2009.

80 Rita Gavriloaie, Wolfgang Nejdl, Daniel Olmedilla, Kent E. Seamons, and Marianne Winslett. No Registration Needed: How to Use Declarative Policies and Negotiation to Access Sensitive Resources on the Semantic Web. In *European Semantic Web Symposium*, volume 3053 of *LNCS*, pages 342–356. Springer, 2004.

81 José Emilio Labra Gayo, Eric Prud'hommeaux, Iovka Boneva, and Dimitris Kontokostas. *Validating RDF Data*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2017. doi:10.2200/S00786ED1V01Y201707WBE016.

82 Qingqing Ge, Zeyuan Zhao, Yiding Liu, Anfeng Cheng, Xiang Li, Shuaiqiang Wang, and Dawei Yin. PSP: pre-training and structure prompt tuning for graph neural networks. In Albert Bifet, Jesse Davis, Tomas Krilavicius, Meelis Kull, Eirini Ntoutsi, and Indre Zliobaite, editors, *Machine Learning and Knowledge Discovery in Databases. Research Track - European Conference, ECML PKDD 2024, Vilnius, Lithuania, September 9-13, 2024, Proceedings, Part V*, volume 14945 of *Lecture Notes in Computer Science*, pages 423–439. Springer, 2024. doi:10.1007/978-3-031-70362-1_25.

83 Floris Geerts, Thomas Unger, Grigoris Karvounarakis, Irini Fundulaki, and Vassilis Christophides. Algebraic structures for capturing the provenance of SPARQL queries. *J. ACM*, 63(1):7:1–7:63, 2016. doi:10.1145/2810037.

84 H. Glaser, A. Jaffri, and I. Millard. Managing co-reference on the semantic web. In *WWW2009 Workshop: Linked Data on the Web*, 2009.

85 Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: An OWL 2 reasoner. *J. Autom. Reasoning*, 53(3):245–269, 2014. doi:10.1007/s10817-014-9305-1.

86 Birte Glimm, Yevgeny Kazakov, Ilianna Kollia, and Giorgos B. Stamou. Lower and upper bounds for SPARQL queries over OWL ontologies. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 109–115. AAAI Press, 2015. doi:10.1609/aaai.v29i1.9192.

87 Birte Glimm and Heiner Stuckenschmidt. 15 years of semantic web: An incomplete survey. *KI*, 30(2):117–130, 2016. doi:10.1007/s13218-016-0424-1.

88 Asunción Gómez-Pérez, Mariano Férnández López, and Oscar Corcho. *Ontological engineering*. Springer, 2004.

89 M. Gori, G. Monfardini, and F. Scarselli. A new model for learning in graph domains. In *Proceedings. 2005 IEEE International Joint Conference on Neural Networks, 2005.*, volume 2, pages 729–734 vol. 2, 2005. doi:10.1109/IJCNN.2005.1555942.

90 Thomas Gottron, Malte Knauf, and Ansgar Scherp. Analysis of schema structures in the linked open data graph based on unique subject uris, pay-level domains, and vocabulary usage. *Distributed Parallel Databases*, 33(4):515–553, 2015. URL: https://doi.org/10.1007/s10619-014-7143-0, doi:10.1007/S10619-014-7143-0.

91 David Gries and Fred B. Schneider. *A Logical Approach to Discrete Math.* Texts and Monographs in Computer Science. Springer, 1993. doi:10.1007/978-1-4757-3837-7.

92 Gerd Gröner, Ansgar Scherp, and Steffen Staab. Semantic web. In Günther Görz, Josef Schneeberger, and Ute Schmid, editors, *Handbuch der Künstlichen Intelligenz, 5. Auflage*, pages 585–612. Oldenbourg Wissenschaftsverlag, 2013. doi:10.1524/9783486719796.585.

93 Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In Balaji Krishnapuram, Mohak Shah, Alexander J. Smola, Charu C. Aggarwal, Dou Shen, and Rajeev Rastogi, editors, *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*, pages 855–864. ACM, 2016. doi:10.1145/2939672.2939754.

94 Andrey Gubichev and Thomas Neumann. Exploiting the query structure for efficient join ordering in SPARQL queries. In *EDBT*, pages 439–450. OpenProceedings.org, 2014.

95 Ivan Habernal and Miloslav Konopík. SWSNL: semantic web search using natural language. *Expert Syst. Appl.*, 40(9):3649–3664, 2013. doi:10.1016/j.eswa.2012.12.070.

96 Harry Halpin and Valentina Presutti. An ontology of resources: Solving the identity crisis. In *European Semantic Web Conference*, pages 521–534, 2009.

97 William L. Hamilton. *Graph Representation Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2020. doi:10.2200/S01045ED1V01Y202009AIM046.

98 William L. Hamilton. *Graph Representation Learning*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers, 2020. doi:10.2200/S01045ED1V01Y202009AIM046.

99 Emil Riis Hansen, Matteo Lissandrini, Agneta Ghose, Søren Løkke, Christian Thomsen, and Katja Hose. Transparent Integration and Sharing of Life Cycle Sustainability Data with Provenance. In *ISWC*, volume 12507 of *Lecture Notes in Computer Science*, pages 378–394. Springer, 2020.

100 Andreas Harth, Katja Hose, and Ralf Schenkel, editors. *Linked Data Management.* Chapman and Hall/CRC, 2014.

101 Olaf Hartig. Zero-Knowledge Query Planning for an Iterator Implementation of Link Traversal Based Query Execution. In *ESWC (1)*, volume 6643 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2011.

102 Olaf Hartig and Carlos Buil Aranda. brtpf: Bindings-restricted triple pattern fragments (extended preprint). *CoRR*, abs/1608.08148, 2016.

103 Olaf Hartig, Katja Hose, and Juan F. Sequeda. Linked Data Management. In *Encyclopedia of Big Data Technologies.* Springer, 2019.

104 Jochen Heinsohn, Daniel Kudenko, Bernhard Nebel, and Hans-Jürgen Profitlich. An Empirical Analysis of Terminological Representation Systems. *Artif. Intell.*, 68(2):367–397, 1994.

105 Lars Heling and Maribel Acosta. Federated SPARQL Query Processing over Heterogeneous Linked Data Fragments. In *WWW*, pages 1047–1057. ACM, 2022.

106 Daniel Hernández, Luis Galárraga, and Katja Hose. Computing How-Provenance for SPARQL Queries via Query Rewriting. *Proc. VLDB Endow.*, 14(13):3389–3401, 2021. URL: `http://www.vldb.org/pvldb/vol14/p3389-galarraga.pdf`, `doi:10.14778/3484224.3484235`.

107 Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan F. Sequeda, Steffen Staab, and Antoine Zimmermann. Knowledge graphs. *CoRR*, abs/2003.02320, 2020. URL: `https://arxiv.org/abs/2003.02320`, `arXiv:2003.02320`.

108 Aidan Hogan, Eva Blomqvist, Michael Cochez, Claudia d'Amato, Gerard de Melo, Claudio Gutierrez, Sabrina Kirrane, José Emilio Labra Gayo, Roberto Navigli, Sebastian Neumaier, Axel-Cyrille Ngonga Ngomo, Axel Polleres, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann. *Knowledge Graphs.* Synthesis Lectures on Data, Semantics, and Knowledge. Morgan & Claypool Publishers, 2021. `doi:10.2200/S01125ED1V01Y202109DSK022`.

109 Ian Horrocks and Peter F. Patel-Schneider. KR and reasoning on the semantic web: OWL. In John Domingue, Dieter Fensel, and James A. Hendler, editors, *Handbook of Semantic Web Technologies*, pages 365–398. Springer, 2011. `doi:10.1007/978-3-540-92913-0\_9`.

110 Katja Hose. Knowledge engineering in the era of artificial intelligence. In *ADBIS*, volume 13985 of *Lecture Notes in Computer Science*, pages 3–15. Springer, 2023.

111 Katja Hose and Ralf Schenkel. Towards benefit-based RDF source selection for SPARQL queries. In *SWIM*, page 2. ACM, 2012.

112 Katja Hose, Ralf Schenkel, Martin Theobald, and Gerhard Weikum. Database Foundations for Scalable RDF Processing. In *Reasoning Web*, volume 6848 of *Lecture Notes in Computer Science*, pages 202–249. Springer, 2011.

113 Yu Hou, Jeremy Yeung, Hua Xu, Chang Su, Fei Wang, and Rui Zhang. From answers to insights: Unveiling the strengths and limitations of ChatGPT and biomedical knowledge graphs. *medRxiv*, June 2023.

114 Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. Graph-mlp: Node classification without message passing in graph. *CoRR*, abs/2106.04051, 2021. URL: `https://arxiv.org/abs/2106.04051`, `arXiv:2106.04051`.

115 Qian Huang, Hongyu Ren, Peng Chen, Gregor Krzmanc, Daniel Zeng, Percy Liang, and Jure Leskovec. PRODIGY: enabling in-context learning over graphs. In Alice Oh, Tristan Naumann, Amir Globerson, Kate Saenko, Moritz Hardt, and Sergey Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023, New Orleans, LA, USA, December 10 - 16, 2023*, 2023. URL: `http://papers.nips.cc/paper_files/paper/2023/hash/34dce0dc3121951dd0399ba02c0f0d06-Abstract-Conference.html`.

116 Toshihide Ibaraki and Tiko Kameda. On the optimal nesting order for computing n-relational joins. *ACM Trans. Database Syst.*, 9(3):482–502, 1984.

117 Dilshod Ibragimov, Katja Hose, Torben Bach Pedersen, and Esteban Zimányi. Processing Aggregate Queries in a Federation of SPARQL Endpoints. In *ESWC*, volume 9088 of *Lecture Notes in Computer Science*, pages 269–285. Springer, 2015.

118 Enrique Iglesias, Samaneh Jozashoori, David Chaves-Fraga, Diego Collarana, and Maria-Esther Vidal. Sdm-rdfizer: An RML interpreter for the efficient creation of RDF knowledge graphs. volume abs/2008.07176, 2020. URL: `https://arxiv.org/abs/2008.07176`.

119 Gautier Izacard, Patrick S. H. Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. Atlas: Few-shot learning with retrieval augmented language models. *J. Mach. Learn. Res.*, 24:251:1–251:43, 2023. URL: `https://jmlr.org/papers/v24/23-0037.html`.

120 Maciej Janik, Ansgar Scherp, and Steffen Staab. The Semantic Web: Collective Intelligence on the Web. *Informatik Spektrum*, 34(5):469–483, 2011.

121 Simon Jupp, Sean Bechhofer, and Robert Stevens. SKOS with OWL: don't be full-ish! In Catherine Dolbear, Alan Ruttenberg, and Ulrike Sattler, editors, *Proceedings of the Fifth OWLED Workshop on OWL: Experiences and Directions, collocated with the 7th International Semantic Web Conference (ISWC-2008), Karlsruhe, Germany, October 26-27, 2008*, volume 432 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2008. URL: `https://ceur-ws.org/Vol-432/owled2008eu_submission_22.pdf`.

122 Pallika Kanani, Andrew McCallum, and Chris Pal. Improving author coreference by resource-bounded information gathering from the web. In *Conference on Artifical intelligence.* Morgan Kaufmann, 2007.

123 Gjergji Kasneci, Shady Elbassuoni, and Gerhard Weikum. MING: Mining Informative Entity Rela-

tionship Subgraphs. In *Information and Knowledge Management*. ACM, 2009.

124 Andreas Kasten. *Secure semantic web data management: confidentiality, integrity, and compliant availability in open and distributed networks*. PhD thesis, University of Koblenz and Landau, Germany, 2016. URL: https://kola.opus.hbz-nrw.de/frontdoor/index/index/docId/1393.

125 Andreas Kasten, Ansgar Scherp, and Peter Schauß. A framework for iterative signing of graph data on the web. In *Extended Semntic Web Conference*. Springer, 2014.

126 Ilkcan Keles and Katja Hose. Skyline Queries over Knowledge Graphs. In *ISWC*, volume 11778 of *Lecture Notes in Computer Science*, pages 293–310. Springer, 2019.

127 Ankesh Khandelwal, Ian Jacobi, and Lalana Kagal. Linked rules: Principles for rule reuse on the web. In Sebastian Rudolph and Claudio Gutierrez, editors, *Web Reasoning and Rule Systems - 5th International Conference, RR 2011, Galway, Ireland, August 29-30, 2011. Proceedings*, volume 6902 of *Lecture Notes in Computer Science*, pages 108–123. Springer, 2011. doi:10.1007/978-3-642-23580-1_9.

128 Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL: https://openreview.net/forum?id=SJU4ayYgl.

129 Holger Knublauch and Dimitris Kontokostas. Shapes constraint language (shacl). W3C Recommendation, 2017. URL: https://www.w3.org/TR/2017/REC-shacl-20170720/.

130 Lecheng Kong, Jiarui Feng, Hao Liu, Chengsong Huang, Jiaxin Huang, Yixin Chen, and Muhan Zhang. GOFA: A generative one-for-all model for joint graph language modeling. *CoRR*, abs/2407.09709, 2024. URL: https://doi.org/10.48550/arXiv.2407.09709, arXiv:2407.09709, doi:10.48550/ARXIV.2407.09709.

131 Christoph Lange, Jörg Langkau, and Sebastian R. Bader. The IDS information model: A semantic vocabulary for sovereign data exchange. In Boris Otto, Michael ten Hompel, and Stefan Wrobel, editors, *Designing Data Spaces: The Ecosystem Approach to Competitive Advantage*, pages 111–127. Springer, 2022. doi:10.1007/978-3-030-93975-5_7.

132 Georg Lausen, Michael Meier, and Michael Schmidt. Sparqling constraints for RDF. In Alfons Kemper, Patrick Valduriez, Noureddine Mouaddib, Jens Teubner, Mokrane Bouzeghoub, Volker Markl, Laurent Amsaleg, and Ioana Manolescu, editors, *EDBT 2008, 11th International Conference on Extending Database Technology, Nantes, France, March 25-29, 2008, Proceedings*, volume 261 of *ACM International Conference Proceeding Series*, pages 499–509. ACM, 2008. doi:10.1145/1353343.1353404.

133 Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the Twenty-first ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, June 3-5, Madison, Wisconsin, USA*, pages 233–246. ACM, 2002. doi:10.1145/543613.543644.

134 Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. Learning entity and relation embeddings for knowledge graph completion. In Blai Bonet and Sven Koenig, editors, *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA*, pages 2181–2187. AAAI Press, 2015. URL: https://doi.org/10.1609/aaai.v29i1.9491, doi:10.1609/AAAI.V29I1.9491.

135 Hao Liu, Jiarui Feng, Lecheng Kong, Ningyue Liang, Dacheng Tao, Yixin Chen, and Muhan Zhang. One for all: Towards training one graph model for all classification tasks. In *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. URL: https://openreview.net/forum?id=4IT2pgc9v6.

136 Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S. Yu, and Chuan Shi. Towards graph foundation models: A survey and beyond. *CoRR*, abs/2310.11829, 2023. URL: https://doi.org/10.48550/arXiv.2310.11829, arXiv:2310.11829, doi:10.48550/ARXIV.2310.11829.

137 Jiawei Liu, Cheng Yang, Zhiyuan Lu, Junze Chen, Yibo Li, Mengmei Zhang, Ting Bai, Yuan Fang, Lichao Sun, Philip S. Yu, and Chuan Shi. Towards graph foundation models: A survey and beyond. *CoRR*, abs/2310.11829, 2023. URL: https://doi.org/10.48550/arXiv.2310.11829, arXiv:2310.11829, doi:10.48550/ARXIV.2310.11829.

138 Alberto Moya Loustaunau and Aidan Hogan. Predicting SPARQL query dynamics. In *K-CAP*, pages 161–168. ACM, 2021.

139 Yuankai Luo, Lei Shi, and Xiao-Ming Wu. Classic gnns are strong baselines: Reassessing gnns for node classification. *CoRR*, abs/2406.08993, 2024. URL: https://doi.org/10.48550/arXiv.2406.08993, arXiv:2406.08993, doi:10.48550/ARXIV.2406.08993.

140 Yao Ma, Xiaorui Liu, Neil Shah, and Jiliang Tang. Is homophily a necessity for graph neural networks? In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. URL: https://openreview.net/forum?id=ucASPPD9GKN.

141 Mohamed Nadjib Mami, Damien Graux, Simon Scerri, Hajira Jabeen, Sören Auer, and Jens Lehmann. Squerall: Virtual ontology-based access to heterogeneous and large data sources. In *International Semantic Web Conference*, pages 229–245. Springer, 2019.

142 Deborah L. McGuinness. Ontologies come of age. In Dieter Fensel, James A. Hendler, Henry Lieberman, and Wolfgang Wahlster, editors, *Spinning the Semantic Web: Bringing the World Wide Web to Its Full Potential [outcome of a Dagstuhl seminar]*, pages 171–194. MIT Press, 2003.

143 Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q.

Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States*, pages 3111–3119, 2013. URL: `https://proceedings.neurips.cc/paper/2013/hash/9aa42b31882ec039965f3c4923ce901b-Abstract.html`.

144 Thomas Minier, Hala Skaf-Molli, and Pascal Molli. Sage : préemption web pour les services publics d'évaluation de requêtes SPARQL. In Nathalie Hernandez, editor, *IC 2019: 30es Journées francophones d'Ingénierie des Connaissances (Proceedings of the 30th French Knowledge Engineering Conference), Toulouse, France, July 2-4, 2019*, page 141, 2019.

145 Fedor Moiseev, Zhe Dong, Enrique Alfonseca, and Martin Jaggi. SKILL: structured knowledge infusion for large language models. In Marine Carpuat, Marie-Catherine de Marneffe, and Iván Vladimir Meza Ruíz, editors, *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL 2022, Seattle, WA, United States, July 10-15, 2022*, pages 1581–1588. Association for Computational Linguistics, 2022. URL: `https://doi.org/10.18653/v1/2022.naacl-main.113`, `doi:10.18653/V1/2022.NAACL-MAIN.113`.

146 Gabriela Montoya, Christian Aebeloe, and Katja Hose. Towards Efficient Query Processing over Heterogeneous RDF Interfaces. In *ISWC (Best Workshop Papers)*, volume 36 of *Studies on the Semantic Web*, pages 39–53. IOS Press, 2018.

147 Gabriela Montoya, Ilkcan Keles, and Katja Hose. Analysis of the Effect of Query Shapes on Performance over LDF Interfaces. In *QuWeDa@ISWC*, volume 2496 of *CEUR Workshop Proceedings*, pages 51–66. CEUR-WS.org, 2019.

148 Gabriela Montoya, Hala Skaf-Molli, and Katja Hose. The Odyssey Approach for Optimizing Federated SPARQL Queries. In *ISWC*, volume 10587 of *Lecture Notes in Computer Science*, pages 471–489. Springer, 2017.

149 Boris Motik, Ian Horrocks, and Ulrike Sattler. Adding integrity constraints to owl. In *OWLED 2007 - OWL: Experiences and Directions*, volume 258, Aachen, 2007. CEUR Workshop Proceedings (CEUR-WS.org).

150 Boris Motik, Ian Horrocks, and Ulrike Sattler. Bridging the gap between owl and relational databases. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(2):74–89, 2009.

151 Boris Motik, Ulrike Sattler, and Rudi Studer. Query answering for OWL-DL with rules. In *International Semantic Web Conference*. Springer, 2004.

152 Kevin P. Murphy. *Probabilistic Machine Learning: An introduction*. MIT Press, 2022. URL: `http://probml.github.io/book1`.

153 Thomas Neumann and Gerhard Weikum. RDF-3X: a risc-style engine for RDF. *Proc. VLDB Endow.*, 1(1):647–659, 2008. `doi:10.14778/1453856.1453927`.

154 Natalya Fridman Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale knowledge graphs: lessons and challenges. *Commun. ACM*, 62(8):36–43, 2019. `doi:10.1145/3331166`.

155 Natasha Noy, Yuqing Gao, Anshu Jain, Anant Narayanan, Alan Patterson, and Jamie Taylor. Industry-scale Knowledge Graphs: Lessons and Challenges. *Communications of the ACM*, 62 (8):36–43, 2019. URL: `https://cacm.acm.org/magazines/2019/8/238342-industry-scale-knowledge-graphs/fulltext`.

156 Daniel Oberle. *Semantic Management of Middleware*. Springer, 2006.

157 Daniel Oberle, Anupriya Ankolekar, Pascal Hitzler, Philipp Cimiano, Michael Sintek, Malte Kiesel, Babak Mougouie, Stephan Baumann, Shankar Vembu, Massimo Romanelli, Paul Buitelaar, Ralf Engel, Daniel Sonntag, Norbert Reithinger, Berenike Loos, Hans-Peter Zorn, Vanessa Micelli, Robert Porzel, Christian Schmidt, Moritz Weiten, Felix Burkhardt, and Jianshen Zhou. DOLCE ergo SUMO: On foundational and domain models in the SmartWeb Integrated Ontology (SWIntO). *Web Semant.*, 5(3):156–174, September 2007. URL: `http://dx.doi.org/10.1016/j.websem.2007.06.002`, `doi:10.1016/j.websem.2007.06.002`.

158 Daniel Oberle, Nicola Guarino, and Steffen Staab. What is an ontology? In Steffen Staab and Ruder Studer, editors, *Handbook on Ontologies*. Springer, 2nd edition, 2009.

159 Sitt Min Oo, Gerald Haesendonck, Ben De Meester, and Anastasia Dimou. Rmlstreamer-siso: an rdf stream generator from streaming heterogeneous data. In *International Semantic Web Conference*, pages 697–713. Springer, 2022.

160 OpenAI. GPT-4 technical report. *CoRR*, abs/2303.08774, 2023. URL: `https://doi.org/10.48550/arXiv.2303.08774`, `arXiv:2303.08774`, `doi:10.48550/ARXIV.2303.08774`.

161 Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. Unifying large language models and knowledge graphs: A roadmap. *CoRR*, abs/2306.08302, 2023. `doi:10.48550/arXiv.2306.08302`.

162 Yannis Papakonstantinou, Hector Garcia-Molina, and Jennifer Widom. Object Exchange Across Heterogeneous Information Sources. In *Data Engineering*, pages 251–260, Washington, DC, USA, 1995. IEEE Computer Society.

163 Jorge Pérez, Marcelo Arenas, and Claudio Gutierrez. Semantics and complexity of SPARQL. *ACM Trans. Database Syst.*, 34(3):16:1–16:45, 2009. `doi:10.1145/1567274.1567278`.

164 Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk: online learning of social representations. In Sofus A. Macskassy, Claudia Perlich, Jure Leskovec, Wei Wang, and Rayid Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 701–710. ACM, 2014. `doi:10.1145/2623330.2623732`.

165 Moritz Plenz and Anette Frank. Graph language models. In Lun-Wei Ku, Andre Martins, and Vivek Srikumar, editors, *Proceedings of the 62nd*

Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024, Bangkok, Thailand, August 11-16, 2024, pages 4477–4494. Association for Computational Linguistics, 2024. URL: https://doi.org/10.18653/v1/2024.acl-long.245, doi:10.18653/V1/2024.ACL-LONG.245.

166 Freddy Priyatna, Óscar Corcho, and Juan F. Sequeda. Formalisation and experiences of r2rml-based SPARQL to SQL query translation using morph. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014*, pages 479–490. ACM, 2014. doi:10.1145/2566486.2567981.

167 Eric Prudhommeaux, Iovka Boneva, Jose Emilio Labra Gayo, and Gregg Kellogg. Shex - shape expressions, 2018. URL: https://shex.io/shex-semantics/index.html.

168 Kashif Rabbani, Matteo Lissandrini, and Katja Hose. Optimizing SPARQL Queries using Shape Statistics. In *EDBT*, pages 505–510. OpenProceedings.org, 2021.

169 Kashif Rabbani, Matteo Lissandrini, and Katja Hose. SHACL and ShEx in the Wild: A Community Survey on Validating Shapes Generation and Adoption. In *WWW (Companion Volume)*, pages 260–263. ACM, 2022.

170 Kashif Rabbani, Matteo Lissandrini, and Katja Hose. Extraction of Validating Shapes from very large Knowledge Graphs. *Proc. VLDB Endow.*, 16(5):1023–1032, 2023.

171 Kashif Rabbani, Matteo Lissandrini, and Katja Hose. SHACTOR: Improving the Quality of Large-Scale Knowledge Graphs with Validating Shapes. In *SIGMOD Conference Companion*, pages 151–154. ACM, 2023.

172 Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67, 2020. URL: https://jmlr.org/papers/v21/20-074.html.

173 Yves Raimond, Christopher Sutton, and Mark B. Sandler. Interlinking Music-Related Data on the Web. *IEEE MultiMedia*, 16(2):52–63, 2009.

174 Steffen Rendle and Lars Schmidt-Thieme. Object identification with constraints. In *International Conference on Data Mining*. IEEE, 2006.

175 Petar Ristoski and Heiko Paulheim. Rdf2vec: RDF graph embeddings for data mining. In Paul Groth, Elena Simperl, Alasdair J. G. Gray, Marta Sabou, Markus Krötzsch, Freddy Lécué, Fabian Flöck, and Yolanda Gil, editors, *The Semantic Web - ISWC 2016 - 15th International Semantic Web Conference, Kobe, Japan, October 17-21, 2016, Proceedings, Part I*, volume 9981 of *Lecture Notes in Computer Science*, pages 498–514, 2016. doi:10.1007/978-3-319-46523-4_30.

176 Philipp D. Rohde. SHACL constraint validation during SPARQL query processing. In Philip A. Bernstein and Tilmann Rabl, editors, *Proceedings of the VLDB 2021 PhD Workshop co-located with the 47th International Conference on Very Large Databases (VLDB 2021), Copenhagen, Denmark, August 16, 2021*, volume 2971 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

177 Philipp D. Rohde and Maria-Esther Vidal. Towards certified distributed query processing. In *Joint Proceedings of the ESWC 2023 Workshops and Tutorials co-located with 20th European Semantic Web Conference (ESWC 2023), Hersonissos, Greece, May 28-29, 2023*, volume 3443 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2023.

178 Carsten Saathoff and Ansgar Scherp. Unlocking the semantics of multimedia presentations in the web with the multimedia metadata ontology. In *International Conference on World Wide Web*. ACM, 2010.

179 Siddhartha Sahu, Amine Mhedhbi, Semih Salihoglu, Jimmy Lin, and M. Tamer Özsu. The ubiquity of large graphs and surprising challenges of graph processing: extended survey. *VLDB J.*, 29(2-3):595–618, 2020. URL: https://doi.org/10.1007/s00778-019-00548-x, doi:10.1007/S00778-019-00548-X.

180 Sherif Sakr, Angela Bonifati, Hannes Voigt, Alexandru Iosup, Khaled Ammar, Renzo Angles, Walid G. Aref, Marcelo Arenas, Maciej Besta, Peter A. Boncz, Khuzaima Daudjee, Emanuele Della Valle, Stefania Dumbrava, Olaf Hartig, Bernhard Haslhofer, Tim Hegeman, Jan Hidders, Katja Hose, Adriana Iamnitchi, Vasiliki Kalavri, Hugo Kapp, Wim Martens, M. Tamer Özsu, Eric Peukert, Stefan Plantikow, Mohamed Ragab, Matei Ripeanu, Semih Salihoglu, Christian Schulz, Petra Selmer, Juan F. Sequeda, Joshua Shinavier, Gábor Szárnyas, Riccardo Tommasini, Antonino Tumeo, Alexandru Uta, Ana Lucia Varbanescu, Hsiang-Yun Wu, Nikolay Yakovets, Da Yan, and Eiko Yoneki. The future is big graphs: a community view on graph processing systems. *Commun. ACM*, 64(9):62–71, 2021. doi:10.1145/3434642.

181 Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Trans. Neural Networks*, 20(1):61–80, 2009. doi:10.1109/TNN.2008.2005605.

182 Ansgar Scherp, Thomas Franz, Carsten Saathoff, and Steffen Staab. A core ontology on events for representing occurrences in the real world. *Multimedia Tools Appl.*, 58(2):293–331, 2012.

183 Ansgar Scherp and Gerd Gröner. Semantic web. In Günther Görz, Ute Schmid, and Tanya Braun, editors, *Handbuch der Künstlichen Intelligenz, 6. Auflage*, pages 783–816. De Gruyter, 2020. doi:10.1515/9783110659948-018.

184 Ansgar Scherp, Carsten Saathoff, Thomas Franz, and Steffen Staab. Designing core ontologies. *Applied Ontology*, 6(3):177–221, 2011.

185 Wolfgang Scheufele and Guido Moerkotte. On the complexity of generating optimal plans with cross products. In Alberto O. Mendelzon and Z. Meral Özsoyoglu, editors, *Proceedings of the Sixteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, 1997.

186 Michael Sejr Schlichtkrull, Thomas N. Kipf, Peter Bloem, Rianne van den Berg, Ivan Titov, and Max Welling. Modeling relational data with graph convolutional networks. In Aldo Gangemi, Roberto Navigli, Maria-Esther Vidal, Pascal Hitzler, Raphaël Troncy, Laura Hollink, Anna Tordai, and Mehwish Alam, editors, *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 593–607. Springer, 2018. doi:10.1007/978-3-319-93417-4_38.

187 Falko Schönteich, Andreas Kasten, and Ansgar Scherp. A pattern-based core ontology for product lifecycle management based on DUL. In *Workshop on Ontology Design and Patterns*, volume 2195 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2018. URL: http://ceur-ws.org/Vol-2195.

188 Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. Fedx: Optimization techniques for federated query processing on linked data. In *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.

189 Juan F. Sequeda and Daniel P. Miranker. Ultrawrap: SPARQL execution on relational data. *J. Web Semant.*, 22:19–39, 2013.

190 Oleksandr Shchur, Maximilian Mumme, Aleksandar Bojchevski, and Stephan Günnemann. Pitfalls of graph neural network evaluation. *CoRR*, abs/1811.05868, 2018. URL: http://arxiv.org/abs/1811.05868, arXiv:1811.05868.

191 Umutcan Şimşek, Elias Kärle, and Dieter Fensel. RocketRML-A NodeJS implementation of a usecase specific RML mapper. In *Proceeding of the First International Workshop on Knowledge Graph Building*, 2019.

192 Manu Sporny, Amy Guy, Markus Sabadello, Drummond Reed, Manu Sporny, Dave Longley, Markus Sabadello, Drummond Reed, Orie Steele, and Christopher Allen. Decentralized identifiers (DIDs) v1.0: Core architecture, data model, and representations. https://www.w3.org/TR/did-core/, 2012.

193 Markus Stocker, Andy Seaborne, Abraham Bernstein, Christoph Kiefer, and Dave Reynolds. SPARQL basic graph pattern optimization using selectivity estimation. In Jinpeng Huai, Robin Chen, Hsiao-Wuen Hon, Yunhao Liu, Wei-Ying Ma, Andrew Tomkins, and Xiaodong Zhang, editors, *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*, pages 595–604. ACM, 2008. doi:10.1145/1367497.1367578.

194 Giorgos Stoilos, Giorgos B. Stamou, Jeff Z. Pan, Vassilis Tzouvaras, and Ian Horrocks. Reasoning with Very Expressive Fuzzy Description Logics. *J. Artif. Intell. Res.*, 30:273–320, 2007.

195 Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: a core of semantic knowledge. In *International Conference on World Wide Web*. ACM, 2007.

196 Kai Sun, Yifan Ethan Xu, Hanwen Zha, Yue Liu, and Xin Luna Dong. Head-to-tail: How knowledgeable are large language models (llm)? A.K.A. will llms replace knowledge graphs? *CoRR*, abs/2308.10168, 2023. doi:10.48550/arXiv.2308.10168.

197 Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. Rotate: Knowledge graph embedding by relational rotation in complex space. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. URL: https://openreview.net/forum?id=HkgEQnRqYQ.

198 York Sure, Michael Erdmann, Juergen Angele, Steffen Staab, Rudi Studer, and Dirk Wenke. OntoEdit: Collaborative Ontology Development for the Semantic Web. In *International Semantic Web Conference*. Springer, 2002.

199 Thomas Pellissier Tanon, Denny Vrandecic, Sebastian Schaffert, Thomas Steiner, and Lydia Pintscher. From freebase to wikidata: The great migration. In *International Conference on World Wide Web*, 2016.

200 Jiao Tao, Evren Sirin, Jie Bao, and Deborah L. McGuinness. Integrity constraints in OWL. In Maria Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010*, pages 1443–1448. AAAI Press, 2010. doi:10.1609/aaai.v24i1.7525.

201 Milena Trajanoska, Riste Stojanov, and Dimitar Trajanov. Enhancing knowledge graph construction using large language models. *CoRR*, abs/2305.04676, 2023. arXiv:2305.04676, doi:10.48550/arXiv.2305.04676.

202 Despoina Trivela, Giorgos Stoilos, Alexandros Chortaras, and Giorgos B. Stamou. Optimising resolution-based rewriting algorithms for OWL ontologies. *J. Web Semant.*, 33:30–49, 2015. doi:10.1016/j.websem.2015.02.001.

203 Giovanni Tummarello, Richard Cyganiak, Michele Catasta, Szymon Danielczyk, and Stefan Decker. Sig.ma: live views on the Web of Data. In *Semantic Web Challenge 2009 at the 8th International Semantic Web Conference (ISWC2009)*, 2009.

204 Michael Uschold. Ontology and database schema: What's the difference? *Appl. Ontology*, 10(3-4):243–258, 2015. doi:10.3233/AO-150158.

205 Michael Uschold and Michael Grüninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33(4):58–64, 2004. doi:10.1145/1041410.1041420.

206 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008, 2017. URL: https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

207 Petar Velickovic, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 -*

May 3, 2018, Conference Track Proceedings. OpenReview.net, 2018. URL: `https://openreview.net/forum?id=rJXMpikCZ`.

208 Ruben Verborgh, Olaf Hartig, Ben De Meester, Gerald Haesendonck, Laurens De Vocht, Miel Vander Sande, Richard Cyganiak, Pieter Colpaert, Erik Mannens, and Rik Van de Walle. Low-cost queryable linked data through triple pattern fragments. In Matthew Horridge, Marco Rospocher, and Jacco van Ossenbruggen, editors, *Proceedings of the ISWC 2014 Posters & Demonstrations Track a track within the 13th International Semantic Web Conference, ISWC 2014, Riva del Garda, Italy, October 21, 2014*, volume 1272 of *CEUR Workshop Proceedings*, pages 13–16. CEUR-WS.org, 2014.

209 Ruben Verborgh, Miel Vander Sande, Olaf Hartig, Joachim Van Herwegen, Laurens De Vocht, Ben De Meester, Gerald Haesendonck, and Pieter Colpaert. Triple pattern fragments: A low-cost knowledge graph interface for the web. *J. Web Semant.*, 37-38:184–206, 2016. URL: `https://doi.org/10.1016/j.websem.2016.03.003`.

210 Maria-Esther Vidal, Simón Castillo, Maribel Acosta, Gabriela Montoya, and Guillermo Palma. On the selection of SPARQL endpoints to efficiently execute federated SPARQL queries. *Trans. Large Scale Data Knowl. Centered Syst.*, 25:109–149, 2016.

211 Maria-Esther Vidal, Edna Ruckhaus, Tomas Lampo, Amadís Martínez, Javier Sierra, and Axel Polleres. Efficiently joining group patterns in SPARQL queries. In *The Extended Semantic Web Conference, ESWC*, 2010.

212 Denny Vrandecic and Markus Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85, 2014. `doi:10.1145/2629489`.

213 Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. KGAT: knowledge graph attention network for recommendation. In Ankur Teredesai, Vipin Kumar, Ying Li, Rómer Rosales, Evimaria Terzi, and George Karypis, editors, *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2019, Anchorage, AK, USA, August 4-8, 2019*, pages 950–958. ACM, 2019. `doi:10.1145/3292500.3330989`.

214 Michael L. Wick, Aron Culotta, Khashayar Rohanimanesh, and Andrew McCallum. An entity based model for coreference resolution. In *SIAM International Conference on Data Mining*, pages 365–376, 2009.

215 Michael L. Wick, Khashayar Rohanimanesh, Karl Schultz, and Andrew McCallum. A unified approach for schema matching, coreference and canonicalization. In *International Conference on Knowledge Discovery and Data Mining*. ACM, 2008.

216 Gio Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38–49, 1992.

217 Kevin Wilkinson, Craig Sayers, Harumi A. Kuno, and Dave Reynolds. Efficient RDF storage and retrieval in Jena2. In *International Workshop on Semantic Web and Databases*, 2003.

218 David Wood. Reliable and persistent identification of linked data elements. In David Wood, editor, *Linking Enterprise Data*, pages 149–173. Springer, 2010. `doi:10.1007/978-1-4419-7665-9\_8`.

219 Felix Wu, Amauri H. Souza Jr., Tianyi Zhang, Christopher Fifty, Tao Yu, and Kilian Q. Weinberger. Simplifying graph convolutional networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 6861–6871. PMLR, 2019. URL: `http://proceedings.mlr.press/v97/wu19e.html`.

220 Lianghao Xia, Ben Kao, and Chao Huang. Opengraph: Towards open graph foundation models. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, *Findings of the Association for Computational Linguistics: EMNLP 2024, Miami, Florida, USA, November 12-16, 2024*, pages 2365–2379. Association for Computational Linguistics, 2024. URL: `https://aclanthology.org/2024.findings-emnlp.132`.

221 Guohui Xiao, Dag Hovland, Dimitris Bilidas, Martin Rezk, Martin Giese, and Diego Calvanese. Efficient ontology-based data integration with canonical iris. In *The Semantic Web - 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3-7, 2018, Proceedings*, volume 10843 of *Lecture Notes in Computer Science*, pages 697–713. Springer, 2018. `doi:10.1007/978-3-319-93417-4\_45`.

222 Han Xiao, Minlie Huang, and Xiaoyan Zhu. Transg : A generative model for knowledge graph embedding. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016. URL: `https://doi.org/10.18653/v1/p16-1219`, `doi:10.18653/V1/P16-1219`.

223 Zi Ye, Yogan Jaya Kumar, Goh Ong Sing, Fengyan Song, and Junsong Wang. A comprehensive survey of graph neural networks for knowledge graphs. *IEEE Access*, 10:75729–75741, 2022. `doi:10.1109/ACCESS.2022.3191784`.

224 Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. Graph transformer networks. In Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 11960–11970, 2019. URL: `https://proceedings.neurips.cc/paper/2019/hash/9d63484abb477c97640154d40595a3bb-Abstract.html`.

225 Vladimir Zadorozhny, Louiqa Raschid, Maria-Esther Vidal, Tolga Urhan, and Laura Bright. Efficient evaluation of queries in a mediator for websources. In *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data, Madison, Wisconsin, USA, June 3-6, 2002*, pages 85–96, 2002.

226 Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer.

Quality assessment for linked data: A survey. *Semantic Web Journal*, 7(1):63–93, March 2015. URL: http://www.semantic-web-journal.net/system/files/swj556.pdf, doi:10.3233/SW-150175.

227 Jianan Zhao, Hesham Mostafa, Mikhail Galkin, Michael M. Bronstein, Zhaocheng Zhu, and Jian Tang. Graphany: A foundation model for node classification on any graph. *CoRR*, abs/2405.20445, 2024. URL: https://doi.org/10.48550/arXiv.2405.20445, arXiv:2405.20445, doi:10.48550/ARXIV.2405.20445.

228 Yuchen Zhuang, Xiang Chen, Tong Yu, Saayan Mitra, Victor Bursztyn, Ryan A. Rossi, Somdeb Sarkhel, and Chao Zhang. Toolchain*: Efficient action space navigation in large language models with a* search. *CoRR*, abs/2310.13227, 2023. URL: https://doi.org/10.48550/arXiv.2310.13227, arXiv:2310.13227, doi:10.48550/ARXIV.2310.13227.