Web Semantics: Science, Services and Agents on the World Wide Web 81 (2024) 100819

Contents lists available at ScienceDirect

Web Semantics: Science, Services and Agents on the World Wide Web

journal homepage: www.elsevier.com/locate/websem

A survey on semantic data management as intersection of ontology-based data access, semantic modeling and data lakes

Sayed Hoseini [a,b,*], Johannes Theissen-Lipp [b], Christoph Quix [c]

[a] Department of Electrical Engineering and Computer Science, Hochschule Niederrhein University of Applied Sciences, Reinarzstrasse 49, Krefeld, 47805, NRW, Germany

[b] Chair of Databases and Information Systems, RWTH Aachen University, Ahornstraße 55, Aachen, Aachen, 52074, NRW, Germany

[c] Department of Data Science and Artificial Intelligence, Fraunhofer Institute for Applied Information Technology FIT, Schloss Birlinghoven, St. Augustin, 53757, NRW, Germany

ARTICLE INFO

Keywords:
Semantic data management
Semantic web
Big data
Data lakes
Ontology-based data-access

ABSTRACT

In recent years, data lakes emerged as a way to manage large amounts of heterogeneous data for modern data analytics. One way to prevent data lakes from turning into inoperable data swamps is semantic data management. Such approaches propose the linkage of metadata to knowledge graphs based on the Linked Data principles to provide more meaning and semantics to the data in the lake. Such a semantic layer may be utilized not only for data management but also to tackle the problem of data integration from heterogeneous sources, in order to make data access more expressive and interoperable. In this survey, we review recent approaches with a specific focus on the application within data lake systems and scalability to Big Data. We classify the approaches into (i) basic semantic data management, (ii) semantic modeling approaches for enriching metadata in data lakes, and (iii) methods for ontology-based data access. In each category, we cover the main techniques and their background, and compare latest research. Finally, we point out challenges for future work in this research area, which needs a closer integration of Big Data and Semantic Web technologies.

1. Introduction

In today's data-driven world, the efficient management and accessibility of data are paramount for stakeholders across various domains. Data lakes have been proposed to tackle challenges in managing unstructured and structured data sources [1]. Numerous organizations have adopted data lakes, facilitating data science projects by combining data from heterogeneous sources [2–5].

However, the mere collection of data is not sufficient. For data to be truly useful, it must be accompanied by meaningful metadata. This is especially crucial for users with limited domain knowledge or those unfamiliar with the specific data sets. Without such metadata, accurately interpreting the data becomes a challenge. The mapping of raw data from data sources to semantically rich models increases the usability and interpretability of data. Knowledge graphs (KGs), formalized with expressive ontologies, have emerged as a powerful tool for enhancing data usability and interpretation [6,7]. They not only provide semantics to raw data but also facilitate data integration through ontology-based data access (OBDA), thereby making data more accessible and interpretable [8].

In recent years, the advantages of data lake research and practice have been increasingly acknowledged. Early implementations focused on efficiently processing Big Data using distributed, scalable systems, such as Hadoop. Concurrently, the importance of proper metadata and data quality management has also been recognized [9,10]. As a result, various strategies for Semantic Data Management (SDM) in data lakes have been introduced [11,12]. These strategies underscore the necessity of integrating Big Data with Semantic Web technologies in semantic data lakes. Within the Semantic Web, the Resource Description Framework (RDF) and the Web Ontology Language (OWL) are the main languages to represent data as a set of Linked Data items [13]. Importantly, these languages also facilitate the representation of metadata through KGs, enhancing the semantic depth and utility of data lakes. Despite the advancements in Linked Data and Semantic Web technologies, the volume of data managed in Semantic Web applications remains significantly smaller compared to that in Big Data scenarios. Therefore, scaling Semantic Web technologies to accommodate large, diverse data sets presents a substantial challenge for their application within data lakes.

* Corresponding author at: Department of Electrical Engineering and Computer Science, Hochschule Niederrhein University of Applied Sciences, Reinarzstrasse 49, Krefeld, 47805, NRW, Germany.
E-mail address: sayed.hoseini@hs-niederrhein.de (S. Hoseini).

https://doi.org/10.1016/j.websem.2024.100819
Received 15 August 2023; Received in revised form 27 February 2024; Accepted 11 March 2024
Available online 27 April 2024

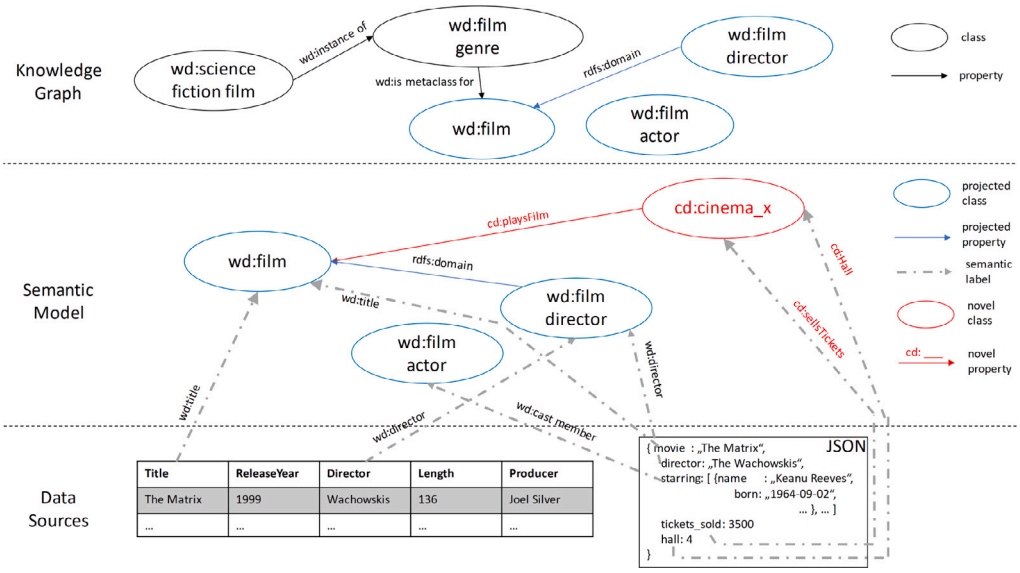

Fig. 1. Semantic data management for heterogeneous data sources in a data lake. Most concepts of the semantic model are drawn from the KG, but to describe specific data sets one may define novel classes and relationships.

Semantic data enriches basic metadata, such as schema and data types, by adding context information not originally present in the data source. Paulus et al. [12] identify two key processes vital for the efficacy and adoption of Semantic Data Management (SDM): (1) **developing and sustaining conceptualizations** and (2) **semantically enriching data sources**, specifically by linking source schemas to these conceptualizations. This semantic enrichment enables users to not only discover data through its conceptual representation but also to comprehend it through the rich context encapsulated within the model.

The process of creating these semantic models is intricate, requiring an analysis of the existing data source, identification and evaluation of relevant Knowledge Graphs (KGs), and the accurate linking of data attributes to KG concepts. This positions the KG as an overarching data model, offering a conceptual description of an organization's data assets. Recognizing the pivotal role of data in driving business processes, companies are increasingly incorporating such models into their data governance and master data management strategies [14]. Modeling a KG demands a significant level of human expertise, as it involves complex decision-making and conceptual mapping. This article focuses on the critical task of aligning data sources with an existing KG, which may be further refined during the semantic modeling phase to ensure a comprehensive representation of the data sources' semantics.

The semantic model acts as a *projection*, mapping the entities and relationships from the KG onto the data sources. This model serves as an intermediary layer, bridging the data layer with the knowledge layer [2]. The basic idea of the semantic model is illustrated in Fig. 1, where raw data sets in the data lake, varying in formats such as tabular or hierarchical JSON, are positioned at the lower data layer and may have overlapping content. The semantic model is represented as a graph comprising nodes and edges borrowed from the KG, further enriched with additional nodes and edges to denote the mapping and augmented semantics of the data sources. For instance, in Fig. 1, we incorporate concepts and properties from the Wikidata knowledge base [15] (prefix *wd*) into our KG. By defining the semantic type and relationships between source attributes in the graph, the semantic model offers a precise description of the data source's intended meaning. A SDM platform should facilitate this process, for example, by recommending relevant concepts and mappings during the model's creation.

Furthermore, the semantic model should be dynamic, allowing for the introduction of new classes and properties as needed. This adaptation becomes necessary when users contribute data sources introducing concepts and relationships not previously covered by the KG. An example provided involves a JSON document about a cinema (using the prefix *cd* for cinema domain) that requires explicit modeling. Such new knowledge must be systematically integrated, thereby enriching the knowledge layer continuously [16].

Semantic labels directly annotate data source elements (e.g., schema attributes) with elements of the semantic model, defined by appropriate predicates and objects. Typically, semantic labels are represented as a triple (schema element, predicate, object). For example, the semantic label for the attribute *Title* in Fig. 1 is articulated through the object *wd:film* and predicate *wd:title*. A semantic data lake should facilitate the derivation of such a semantic model from raw datasets, thus providing a more conceptual data description that includes concepts and their interrelations, and linking data sets to their relevant concepts [14].

In Fig. 2, we introduce a revised data lake architecture [17]. The architecture is structured in four layers: (1) the *ingestion layer* deals with data and metadata extraction; (2) the *storage layer* provides systems to store and query data and metadata in a scalable way; (3) the *transformation layer* enables data transformation and integration; (4) the *interaction layer* offers exploration and query functions to users. Each layer has been extended with enhanced metadata functions with semantic capabilities. For instance, the ingestion layer's semantic labeling component assigns semantic labels to metadata elements. Data Quality (DQ) management benefits from verifying the presence of semantic information for data sources. Managed within the storage layer, the semantic information (labels, models, KGs, etc.) in an expanded semantic metadata repository supports functions such as OBDA, aiding in data usage and interpretation. Consequently, the interaction layer incorporates additional features, such as KG and semantic model browsing, semantic query formulation using languages like SPARQL,[1] and tools for refining semantic mappings and models.

1.1. Scope, research methodology and contribution

In this survey, we give an overview of the recent developments in SDM related to **semantic data lakes** (see Definition 2) in particular. To provide a clear and logical presentation, we were inspired by the pipeline presented by [18] (see Fig. 3). It starts with the schema analysis, a standard procedure of any data lake ingestion module including

[1] https://www.w3.org/TR/rdf-sparql-query/

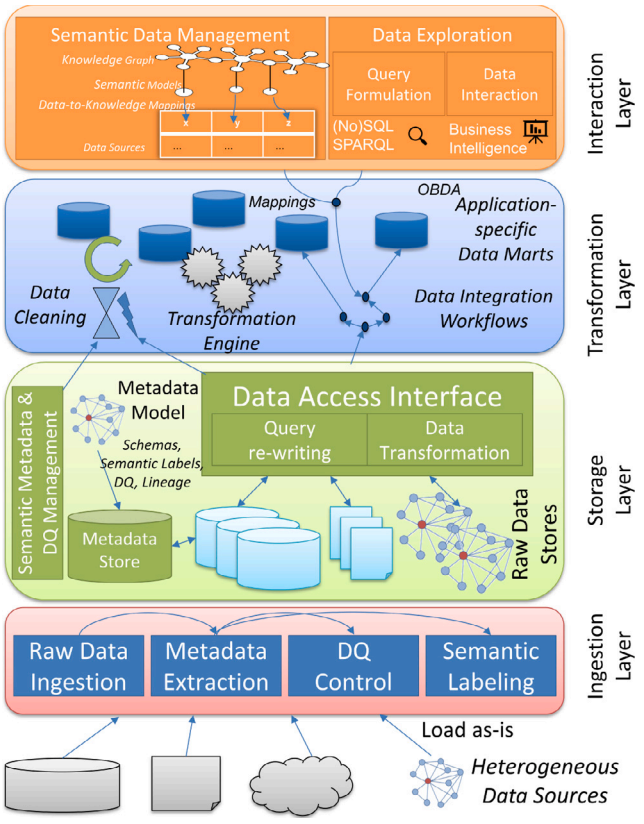

Fig. 2. Semantic data lake architecture, extended from [17].

the ones presented here (see Section 3.1.1 to Section 3.1.3). The next steps are semantic labeling (Section 3.2.1), modeling (Section 3.2.2) as well as refinement and storage (Section 3.2.3). Furthermore, the original figure has been extended by OBDA (Section 4), because it provides a uniform data access interface for heterogeneous data in data lakes based on the semantic descriptions.

To collect the relevant papers, we made use of Google Scholar primarily. We first used a list of keywords, which was manually created when reading relevant papers starting from e.g. [12,14,19]) and evolved dynamically during the writing: 'semantic data lake', 'knowledge graph data lake', 'semantic data catalog', 'semantic data integration', 'semantic/ontology-based data management', 'semantic models', 'semantic annotation', 'semantic table interpretation', 'semantic type detection', 'ontology-based-data-access', 'linked data & data lake'. For each query, we first retrieve relevant results by importing the DOI into *Citavi*, a program for reference management, then read each retrieved paper to judge whether the paper is relevant for one of the three domains of interest (Semantic Data Lakes, Semantic Data Management/Integration, OBDA). From there we assigned each paper to a category resembling the structure of this article. We made heavy use of the *citation snowballing method* [20], i.e., extending the results by searching through the cited references in the initial keyword-based findings as well as exploiting the 'Cited By' functionality in Google Scholar. The commercial systems were found using Google Search.

In summary, this article makes the following contributions:

- A definition of the terms related to SDM is provided in Section 2.1. Terms like semantic models or semantic labels are frequently used in the context of SDM; to make a proper classification and comparison of systems and approaches in this area, we first need clear definitions. Additionally, a taxonomy (Fig. 4) of the various notions is presented as a high-level overview.
- In Section 2.3, we introduce an evaluation framework by deriving several criteria.
- The state-of-the art for SDM is discussed in Section 3. We review metadata models and data lakes addressing semantics in Section 3.1.
- Section 3.2 addresses in detail approaches for semantic modeling, i.e., algorithms for creating the semantic models (semi-)automatically. We distinguish between approaches for semantic labeling and semantic modeling.
- Based on the evaluation framework, we perform a detailed comparison for metadata models and data lakes as well as SDM systems in Section 3.3.
- The most advanced use of semantic information is for ontology-based data access (OBDA). A KG is used as a common data model for specifying queries; OBDA systems take mappings to data sources into account and translate queries into the query language of the particular data source. Especially in this context, scalable approaches are required for query processing over Big Data. We review several systems in this area that have been proposed in recent years in Section 4.
- In Section 5, we discuss two application areas for SDM in data lakes: Industrial Internet-of-Things and Smart Cities. This shows that the SDM techniques can be applied to real-world use cases.
- Finally, we derive in Section 6 challenges that still need to be addressed in the field of semantic data management.

1.2. Related surveys & work

Sawadogo et al. [21] focus on generic data lake architectures and metadata management and discuss the pros and cons of data lakes and their design alternatives. Similarly, Hai et al. [22] present a comprehensive overview of research questions for designing and building general data lakes. They classify the existing approaches and systems based on their provided functions for data lakes and provide a thorough comparison of existing solutions and the discussion of open research challenges. In [23] Adamou et al. describe possible Linked Data use cases in data lakes on a high level including basic graph-based data storage and querying, data integration of internal and external data as well as describing and profiling data sources for data cataloging. While they do not go into much technical detail, they offer a comprehensive perspective on the utility of Linked Data in addressing these functionalities. To illustrate how Linked Data principles and technologies can be applied to data lakes, they provide a practical scenario for establishing a data platform for a smart city. Similarly, Wecel et al. [24] describe how to use the Linked Data for the general enrichment of data assets. Couto et al. [25] conduct a systematic literature review about data integration in data lakes where they identify examples of integration models, how to calculate the similarity among the datasets, how the models are evaluated and the most common type of data to be integrated.

Paulus et al. [12] investigate existing semantic modeling approaches, discuss their strengths and weaknesses for real-world use, and present future challenges and necessary research directions that the community needs to focus on to make semantic data management acceptable in everyday business. Data lakes and scalability to Big Data is not in the focus of Paulus et al.

Xiao et al. present first a more theoretical discussion on the formal implementation details in OBDA [26]. In a follow-up work [8], they present the tooling ecosystem and concrete use cases in a wide range of commercial applications. Additional works in this context and a more detailed discussion can be found in Section 4.

A recent survey by Liu et al. [27] reviews approaches for semantic labeling (see Definition 3) which is closely related to Section 3.2.1 of this work.

In contrast to the mentioned related work in this field, we focus especially on approaches which can be **scaled up to Big Data within a**

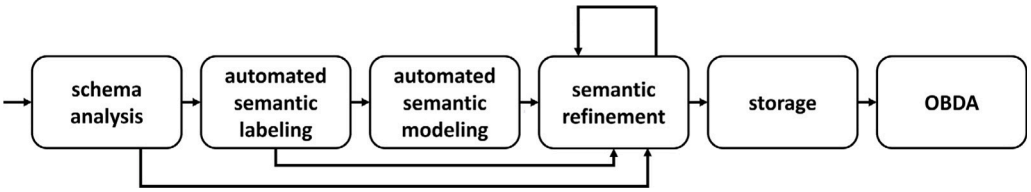

Fig. 3. Overview of the phases during the automated generation of a semantic model as illustrated in [18] extended by OBDA.

data lake and facilitate the **semantic integration of data**. In a holistic approach, we address the complete process as shown in Fig. 3, starting from SDM in data lakes, to semantic modeling for data integration, and finally OBDA. In each section, we focus especially on the applicability to data lakes and Big Data.

2. Semantic data management in data lakes

We will first provide definitions for the main terms used in this survey (Section 2.1) and discuss in detail the distinction between semantic labeling and semantic modeling (Section 2.2). Following this, a taxonomy is presented (Fig. 4), offering a high-level overview of the various concepts, along with a framework (Section 2.3) that will be used to assess the state-of-the-art in the succeeding chapter.

2.1. Definitions

We have been using terms like semantic data lakes, semantic labels, and semantic models in the introduction and now clarify their meaning. All given definitions should be general and are not limited to a specific technology. Semantic tools often apply W3C standards (e.g., RDF, OWL), but some industrial solutions do not focus on compatibility with a standard and model KGs and semantic metadata differently.

Definition 1 (*Data Lake*)**.** A data lake is a data repository that manages heterogeneous data sets in their original structures. It should provide functions to extract data and metadata from heterogeneous sources, manage the data efficiently in a hybrid storage system, and query and transform the data in a scalable way [17,22]

In data warehouses, the classical ETL (Extract-Transform-Load) idea is applied: data is transformed into a uniform schema before it is loaded into the data warehouse. Only then, queries can be executed on this integrated data set. In data lakes, raw data is loaded in a heterogeneous storage layer and then transformed only if required for a specific application or query. Thus, data lakes apply an ELT approach (Extract-Load-Transform). Query processing in data lakes should also be able to deal with heterogeneous data sources, as data is not transformed into a uniform format before it is loaded into the lake. Apache Spark is a system frequently used for the implementation of data lakes as it can process queries over heterogeneous data sources.

This is also an important aspect of the scope of this survey: we only consider approaches that do not require the data to be converted into a uniform schema, as such approaches are just not scalable to Big Data. For example, Chessa et al. [28] describe a general methodology for extending a data lake with a knowledge graph. The method provides that data sources are transformed into RDF using the RML mapping language (see Section 4) using a manually created domain ontology to particular use cases. Even though they claim to be following the ELT approach, in our interpretation, since any data asset in the lake has to undergo a common transformation before being exposed to a consumer, consequently we view this as an ETL approach.

Definition 2 (*Semantic Data Lake*)**.** Semantic data lakes are a specific form of traditional data lakes that extend the capabilities through a *semantic layer* that enriches and connects the stored data *semantically*. The *semantic layer* equips data sets in the lake with connections from the data set's metadata to conceptual/logical models, which encapsulate knowledge potentially external to the content of the data, such as domain knowledge.

Definition 3 (*Semantic Labels*)**.** Semantic labels manifest the connection between data sources and concepts of a conceptualization with the objective of describing the semantics of the data source's attribute to which the label is associated.

Semantic labels are the core concept of implementing semantic data management in practice. They complement extractable metadata (such as data types, sizes, formats, etc.) to convey **context** information that may not be inherent to the data source at hand. The most common semantic labels are those that connect elements of a data source's schema in a one-to-one-relationship, e.g., the column of a table or a leaf node in a hierarchical/nested data set (e.g., JSON file), to their counterparts from a KG as previously described. However, they can also be used for example on the level of individual data values, for a specific row of the table, to denote relationships between individual schema entries, the entire data set, or any other relevant concept. In the literature, various terms are used for semantic labels, for example, semantic enrichment, links, tags, types, profiling, or annotations [21, 29–31]. The idea behind semantic labels differs from approaches that model their entire metadata/data catalog as a KG, such as *Aurum* [32], by the fact that their primary objective is to provide context, hence additional information from external sources, in contrast to the latter which stores metadata in a graph-based data model.

In the area of OBDA (see Section 4), the term mapping is also commonly used. Mappings establish a connection between data attributes and KGs and they are the technical prerequisite for query rewriting, e.g., from SPARQL to SQL. While mappings in OBDA refer to different notions technically, on a conceptual level however, they can be regarded as closely related to semantic labels, because essentially they connect the metadata of a data source to instances of a conceptualization. The main difference is that mappings in OBDA require precise semantics to rewrite queries correctly. Semantic labels are usually not defined with precise semantics in mind, as they aim at providing more context information to the user or at connecting related data sets.

Definition 4 (*Semantic Models*)**.** A semantic model specifies the broader context of a data source by extending the semantic labels through additional meaningful concepts and provides suitable relationships that hold between various concepts.

To make the definitions more clear, we provide a rough formalization, explained by referring to the case study of Fig. 1. We consider two data sources stored in a data lake, a table and a JSON file where each data source D consists of a set of attributes $D = (c_1, \ldots, c_n)$ (i.e., columns in a table or paths in hierarchical data sets, respectively). For the conceptualization, we consider a standard knowledge graph KG defined by a set of RDF triples in the form of subject–predicate–object: (s, p, o).

The idea behind a semantic label for c_i is to best describe the semantics of c_i by defining a triple $SL_{c_i} = (c_i, p, o)$, with p and o preferably, but not necessarily, coming from KG. For example, the semantic label for the JSON key "tickets_sold" is represented as the triple: (tickets_sold, cd:sellsTickets, cd:cinema_x). Here, p and the o are

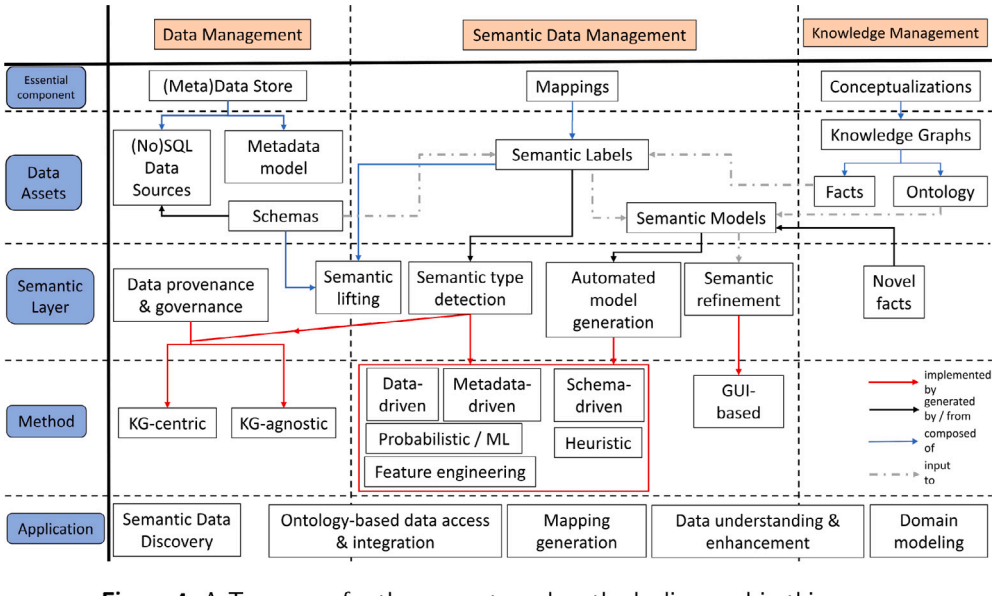

Fig. 4. A Taxonomy for the concepts and methods discussed in this survey.

novel concepts for a specific domain, that are not present in this form in the general-purpose ontology of WikiData. Users may incorporate these novel facts into the overall knowledge layer, thereby evolving it dynamically. On the other hand, the table's column "Title" and the JSON's key "movie" possess the same predicate and object and therefore provide a suitable entry point for data integration between the two heterogeneous data sets.

The semantic model SM is a set of triples built upon the set of semantic labels $SL = \{SL_{c_1}, \ldots, SL_{c_n}\}$. In order to specify the broader context of the data source D, it may additionally contain further nodes and edges drawn from KG denoted as SM_{KG}. The semantic model can be extended by concepts and relationships of the sources, which are not represented in KG. We denote this extension of the semantic model as SM_X. Note that SL, SM_{KG}, and SM_X are pairwise disjoint. $SM = SL \cup SM_{KG} \cup SM_X$

(cd:cinema_x, cd:playsFilm, wd:film) $\in SM_X$

(wd:film director, rdfs:domain, wd:film) $\in SM_{KG}$

("Title", wd:title, wd:Film) $\in SL$

The given case study is restricted to the schema of the two data sources for the purpose of illustration, but the same formalization can be generalized to various scenarios. Naturally, it can easily be applied to other concepts in the data layer, e.g., entire data sets and relationships between them (replace c_i with data set d_i). In Fig. 1, modeling the film genre of the movie "The Matrix" on the level of individual data entries (single row/object or specific cell entry/value in the table/JSON) of the data sources could have been included.

Some approaches, like *DoDuo* [33], establish the relationship between any two columns in a table directly via a predicate without an intermediate element of the semantic model in between. For example, the triple is composed of two table columns and only the predicate comes from the conceptualization: (c_i, p, c_j). Semantic labels, especially when considered as 'weak' annotations, are frequently also modeled as just an additional field in the metadata repository and not expressed as a triple in RDF, e.g., in *GEMMS* [31]. If one thinks of the predicate being isRelatedTo, the given formalization still applies with no further specification.

2.2. Semantic data management taxonomy

In Fig. 4, we have structured the various concepts and methods discussed in this survey into a taxonomy. The approaches are vertically separated by the three main areas addressed in this article: First, there is (big) data management, in which we limit the discussion to data lakes and for which we refer the interested reader to the following articles [22,34]. Then, there is knowledge management, in which we limit the discussion to KGs. Here, we point to the following articles [6,35]. The intersection between the previous two is represented by SDM and this is the focus of this work. We further differentiate five dimensions horizontally which gradually decrease the level of abstraction to culminate into concrete applications which are described throughout the article. The first dimension represents higher-level conceptual components; in this article, these include a data lake system (i.e., (No)SQL sources and their metadata), semantic labels and models, as well as serialized knowledge in the form of KGs in the second dimension. In the third dimension, the semantic layer includes functionalities that a semantic data lake system may implement to generate the semantics required for effective SDM, which are implemented by different methods in the fourth dimension.

Semantic labeling is the process of creating semantic labels for a data source. In the literature, the automation of this process is also denoted as *Semantic Type Detection* (STD) [36] or *Semantic Table Interpretation* (STI) [27]. Burgdorf et al. [37] identify three different directions for this task: *schema-driven*, i.e., using the schema of each data point within a data set, such as [38,39]; *data-driven*, i.e., using the actual data values contained in a data set to assign fitting concepts with statistics or machine-learning-based classification, such as [40–42]; *metadata-driven*, i.e., using all available additional pieces of information on a data set that might contribute to the semantic labeling of the data, e.g., comments or textual data documentation, such as [43]. Liu et al. divide approaches further into ML-based, heuristic, or feature-engineering methods [27].

From the taxonomy, it is evident that the concept of the semantic label, combining facts stored in the KG and schema attributes of the data sources, is central to the whole framework. This is one of the reasons why STDs have gained so much attention recently [27]. Another crucial property is whether the detection can be performed against arbitrary KGs (see Section 3.2). The labels are the prerequisite for the generation of semantic models that are ultimately refined using graphical interfaces and potentially contribute novel facts to the KGs. Data provenance and governance can be enhanced or built entirely from these semantic data structures and again there exists an important distinction between KG-centric and agnostic approaches (see Section 2.3). Finally, there exists a variety of applications, i.e., semantic

data discovery (e.g. semantic similarity [20]) or ontology-based data access (see Section 4).

The Semantic Web Challenge on Tabular Data to Knowledge Graph Matching[2] (*SemTab*) is an annual competition (since 2019) for researchers to present novel approaches evaluated with regards to accuracy and usability on proposed benchmarks. The challenge's problems include three tasks to annotate a table with semantics. Column Type Annotation (CTA) is a schema-level annotation task that directly corresponds to our interpretation of *semantic labeling* as it aims to map the underlying table schema to a KG. Cell Entity Annotation is a cell-based annotation task that deals with annotating the values in the table cells with entities inside a KG. Finally, Column Predicate Annotation (CPA) is a schema-level annotation that defines binary relationships (properties of the KG) between pairs of columns of the table. [27] further consider the tasks of Row-to-instance (R2I), i.e. annotating an entire row of a relational table with a KG entity, and Topic annotation, i.e. annotating the entire table with a concept or an entity from the target KG. Because Semantic Models, OBDA as well and most data integration techniques predominantly live on the level of a data source schema, the only relevant tasks we consider here are CTA & CPA.

With *semantic modeling*, we refer to the process of creating relationships between the semantic labels thereby describing the data source's context. In practice, semantic modeling often depends on accurate semantic labels, which can be viewed as elements of a semantic model of only the first order. In Section 3.2 we discuss recent papers that present algorithms to automate these processes.

Paulus et al. [18] introduce a phase called *semantic refinement*. They reinforce the view that the automated creation of semantic labels and models is a self-evident requirement for any OBDM platform. While automated approaches can provide initial semantic labeling and modeling after analyzing the schemata of the data sources, erroneously chosen or missing labels for concepts will have a severe impact on the semantic modeling phase. Therefore, the human operator shall have the opportunity to improve the quality of the semantic labels as well as the resulting model manually by checking, validating, correcting, selecting, or exchanging ambiguous or mutually exclusive concepts, as well as adding new concepts that were not included initially.

Mami et al. [44] describe a transformation called *semantic lifting*. Given a set of semantic labels and a (yet non-semantic) data set, a semantic lifting function returns a semantically annotated data set with semantic labels for entities and attributes. Mami et al. considered semantic labels only in their original proposal, however, *semantic lifting* can potentially be applied to include entire semantic models as well as any other descriptive metadata (e.g., comments or textual data documentation stored in documentation tools or Wikis). Transforming data sources to incorporate semantic information along with the raw values beforehand to prepare them for subsequent semantic access necessarily corresponds to an ETL approach contradicting the ELT principle that is inherent to data lakes. When dealing with large data sets, this transformation can be challenging and expensive, particularly when data liveliness is important.

2.3. Evaluation framework for SDM

Before the presentation of the State-of-the-Art, we introduce an evaluation framework for the first half of this article dealing with SDM. We identify criteria to compare and to check the comprehensiveness of the existing proposals in the literature:

1. **SL:** *Semantic labels* denotes the possibility of connecting schema entries with instances in an external KG, either a domain KG to be uploaded or an online KG such as schema.org.
2. **SR:** *Semantic relationships*, i.e., any form of hierarchical, generic, or pre-defined semantic relationships (semantic connections between data sets, e.g., for provenance or governance). It should be noted that none of the existing data lake systems provides advanced modeling features such as those depicted in Fig. 1.
3. **C/E:** *Central KG graphs vs external KGs*. Here we distinguish whether the underlying metamodel itself is modeled directly as a KG (i.e., relationships are limited to those defined in this KG). A more flexible system would allow an external KG, and any concepts and relationships from this KG can be used in the semantic model. A checked field indicates that the metadata model uses a central, predefined KG.
4. **MI:** *Metadata interoperability* This denotes whether the authors have considered the possibility of importing or exporting metadata. i.e., serialization of the semantic model. This is strongly related to criterion **C** since serialized models are ported to different systems more easily.
5. **IAL:** *Initial automatic semantic labeling* denotes the functionality to generate semantic labels automatically from the schema at the time of ingestion.
6. **TA:** *Technical abstraction*. This denotes whether the authors have explicitly considered the technical abstraction of the semantic modeling as a requirement to increase usability for unfamiliar users. This means that technical details are hidden from the users, e.g., a user does not need to be familiar with RDF.
7. **C:** *Compatibility with Semantic Web technology*. This criterion checks whether semantic labels are modeled in a language of the Semantic Web (e.g., RDF, OWL).
8. **O:** *Open Source*. This criterion checks whether there is an open-source implementation available.
9. **IAM:** *Initial automatic semantic model*. In contrast to IAL, this denotes here the functionality to generate entire semantic models automatically from the schema after the ingestion of a data source. This includes the first three phases in Fig. 3.
10. **EX:** *Support for external KGs*. Is the approach capable of handling arbitrary KGs or fixed to a predefined KG?
11. **M:** *Maturity*. Here we check whether the system has been reported to be successfully utilized in a Big Data project.
12. **C:** *Compatibility with Semantic Web technology*. This criterion checks whether semantic labels are modeled in a language of the Semantic Web (i.e., RDF or OWL).

Criteria one to eight are applied in Table 1 to concrete data lake systems that do not list approaches that propose a metadata model only. Criteria eight to twelve are applied in Table 2 to modeling platforms, that include methods for semantic labeling and/or modeling, as well as a graphical user interface.

3. Semantic data management

Now, that an evaluation framework is set up, we proceed with illustrating the State-of-the-Art in the field of SDM. We begin with a discussion of the literature on semantic data lakes in Section 3.1, followed by Semantic Modeling for Data Integration in Section 3.2. These next sections serve as the foundation for the discussion and comparison in Section 3.3 based on the evaluation framework.

3.1. Semantic data lakes

Semantic Data Lakes have garnered significant attention in recent years as the primary means for conducting semantic data management in practice. While some researchers focused primarily on theoretical aspects, others proposed practical implementations. We have decided to structure the upcoming discussion into proposals for metadata model (Section 3.1.1), actual systems (Section 3.1.2), and commercial solutions (Section 3.1.3).

[2] https://www.cs.ox.ac.uk/isg/challenges/sem-tab/index.html

Table 1
Comparison of data lakes with a semantic layer. The criteria are defined in Section 2.3.

	SL	SR	C/E	MI	IAL	TA	C	O
Goods[45]	✓	✓	X	X	✓	X	X	X
Constance[9]	✓	X	X	X	X	X	X	X
Aurum[32]	✓	✓	✓	X	✓	X	X	✓
KGLac[46]	X	✓	✓	X	✓	✓	✓	X
CoreKG[47]	✓	✓	✓	✓	✓	✓	✓	✓
BARENTS[48]	X	✓	✓	✓	X	✓	✓	X
Enterprise Knowldege Graph Plattform by Stardog at Bosch	✓	✓	✓	✓	✓	✓	✓	X
Semantic Integrator from the Semantic Web Company	?	✓	?	?	✓	✓	?	X
Anzo Semantic data lake	?	✓	?	?	✓	✓	?	X
Semantic Layer by AtScale	X	✓	X	✓	✓	✓	X	X
Semantic Layer Dremio LakeHouse Plattform	X	✓	X	✓	✓	✓	X	X

Table 2
Comparison of semantic modeling platforms.

	IAM	SR	EX	M	C	O
ESKAPE [2]	✓	X	✓	✓	X	X
KARMA [49]	X	✓	✓	✓	✓	✓
PLASMA [50]	✓	✓	✓	✓	✓	✓
ODIN [51]	✓	✓	✓	X	✓	✓

3.1.1. Metadata models for data lakes that account for semantics

Metadata models for data lakes have been an active area of research in recent years and many existing proposals consider semantics as a fundamental issue for the management of heterogeneous data sources.

MEDAL and its successor *goldMEDAL*, [29] identify requirements for a generic metadata model for data lakes and consider semantic enrichment as a core requirement. The earlier proposals *HANDLE* [52] as well as *GEMMS* [31] enable manual semantic labeling already.

Other approaches model their metadata directly in RDF as an ontology. An early approach is presented by Farid et al. [10], where all source data and metadata are stored in the RDF format. Diamantini et al. [53] choose a network-based and semantics-driven representation for their model that can be extended with external KGs. In a follow-up work [54], they describe another ontology-based metadata model tailored to query-driven analytics. Here, a special focus is laid on key performance indicators and statistical measures typical in data science, where the users specify the indicators of interest at query time, as well as related dimensions of analysis, in terms of KG concepts. The framework then performs the required data discovery and on-demand integration to provide the requested indicators. Diamantini et al. present a quantitative evaluation of their work on semantic data lakes in terms of joinability of data sources in [55].

A similar approach is followed by Bagozi et al. [56] who utilize indicators to enable personalized data lake exploration. Indicators are computed based on the centralization of the data storage, according to a less flexible ETL approach than what traditional data lakes offer. In addition, domain experts, who know the data stored within the data lake, are usually distinct from data analysts, who define indicators, and users, who exploit indicators. Hence, the authors allow domain experts to enrich the heterogeneous sources within a data lake with semantic models using domain ontologies. Further, they propose an ontology used by analysts to define indicators and analysis dimensions, in terms of concepts within semantic models as well as formulas to aggregate them. In [57,58], Bianchini et al. demonstrate the effectiveness of this model in as Smart City environment extended by theoretical considerations about modeling user preferences to enable personalized exploration of data.

At Bosch, the potential of semantic technologies has already been exploited at scale in production [59]. The development of KGs required an immense initial effort by the company but promised a long-term solution for a series of sophisticated challenges. Dibowski et al. [14,60] describe a metadata model represented as an ontology called *DCPAC*, based on W3C recommendations such as the Data Catalogue Vocabulary (DCAT[3]) and the PROV Ontology (PROV-O[4]) for data catalog management and provenance control. Ingested data assets are enriched with semantics by aligning, annotating, and enriching the input data with DCPAC concepts. In addition, they utilize a new concept called ontology-driven, self-adaptive frontends, in which the data catalog's GUI can dynamically render information from a KG. A similar concept was presented in [61]. This is useful because changes in the underlying ontology do not require any changes at the frontend.

3.1.2. Existing semantic data lake systems

There exist several proposals for data lake systems that include functions for SDM.

Google's Dataset Search (Goods) [45] bootstraps its data catalog by crawling Google's storage systems and logs to discover what data sets exist. It was first used internally for private data sets in Google, but in the meantime, it has been released as a research prototype for public data sets [62]. It examines the data's content and metadata and matches it against Google's Knowledge Graph to identify entities. With more than 70 billion assertions describing a billion entities [63], the graph is the result of more than a decade of data contribution activity from a diverse set of individuals that powers APIs for YouTube and Google Cloud. For example, a query for a famous person produces a small panel in the search results with information from the knowledge graph.

Besides its metadata model, *GEMMS* [31] is a tool for automatic metadata extraction in data lakes. The authors recognized that often acronyms or synonyms are used as the name of a schema element. By using semantic annotations, such ambiguities can be avoided and the semantics of elements can be clearly defined. *Constance* [9] takes up this idea and is one of the first data lake systems to enable semantic labels on the level of the data source's schema.

Apache ATLAS is a stand-alone data governance framework for Hadoop that has been successfully integrated into data lakes [64] allowing semantic labeling. *goldMedal* demonstrates their metamodel for data lakes using *Apache ATLAS*. Hence, it is neither viewed as a standalone data lake system nor a metamodel, but may be utilized as a core technology worth being mentioned.

The data discovery system *Aurum* [32] has KGs as a central element. The system builds a so-called enterprise KG in a single pass through the data sources by using data summarization and hashing to capture relationships between them. *Aurum* describes the data discovery problem by properties and relationships of the data sources stored in the data lake. Properties include schema labels and quality measures, relationships may include similarity and primary/foreign key relationships. In the KG, each node represents columns of the data sources, edges represent relationships between two nodes, and hyperedges connect any number of nodes that are hierarchically related, such as columns of the same table, or tables of the same source. *Aurum* allows individual

[3] https://www.w3.org/TR/vocab-dcat-2/
[4] https://www.w3.org/TR/prov-o/

entries to be represented but suggests that the additional storage costs are not offset by the gain in expressiveness. To calculate the weights associated with each property, it considers content similarity (the values of the columns are similar), schema similarity (the attribute names are similar), or the existence of a foreign-key relationship between them. As such, the underlying algorithm to generate the hypergraph involves two steps: firstly, each data source is profiled, and secondly, relationships are calculated based on the profiles. Querying is performed through a language called Source Retrieval Query Language (SRQL) specifically designed for the enterprise KG, which is significantly different from SPARQL.

KGLac [46] follows a similar approach in that it is an external service that generates a central KG based on the extracted profiles to perform all data management. The data profiling is assisted by a machine learning model to represent each data source in an embedding space to perform a similarity search between tables or columns without revealing the raw data. Again, in the KG vertices represent data sources, i.e., tables, or columns, while edges represent relationships between these nodes. In contrast to *Aurum*, KGLac relies on a custom extension of the RDF standard. A specific ontology is used for querying with SPARQL to allow navigation and maintenance of the central KG. The system does not have a user interface but relies on Python APIs to query a RDF database with direct integration into common data science pipelines.

CoreKG [47] is an open-source service developed for data lakes to curate stored data sources via REST APIs, which is composed of the extraction of metadata, their enrichment, linking, and annotation. Linkage in CoreKG means establishing a connection to an external KG such as WikiData as well as to other data sources. It is not a complete data lake, because it does not provide any storage or processing capabilities, but its functionality can be easily added to existing data lakes using the REST API.

BARENTS [48] introduces an ontology-based method that enables analysts to model how data sets have to be pre-processed to transform them into meaningful knowledge. They introduce an ontology, not for data discovery but explicitly for pre-processing. Domain experts can use this ontology to describe which data transformations have to be applied to the raw data from source systems to make it exploitable in their use cases.

3.1.3. Commercial systems

There exists a series of proprietary software stacks worth being mentioned, but due to the 'paywall', the functionality and details are not fully revealed.

The metadata model used by Bosch is implemented using the *Enterprise Knowledge Graph Plattform* by Stardog[5] for storing and processing the semantic layer as a KG. The *Poolparty Semantic Suite*[6] from the Semantic Web Company offers a wide range of services, accessible through (REST) APIs that can be exploited in conjunction with a data lake. The *Anzo Semantic Data Lake*[7] offered by Cambridge Semantics focuses on adding context by first building a catalog of KGs from existing sources in a data lake and then aligning those to semantic models based on business meaning.

The *Semantic Layer* by AtScale[8] follows a different approach in that it does not apply Semantic Web technologies at all. Instead, it focuses on data integration through data modeling in a canvas, where users can establish relationships and hierarchies between heterogeneous data sources from different storages without ever replicating or moving any data, but only via reference. The *Semantic Layer* complements cloud data platforms (such as DataBrick's *Lakehouse* [65]) and provides a platform for integration and orchestration of the platform's query and transformation engines. This bridges the gap from the data sources to business intelligence tools and AI applications. A typical workflow involves selecting data sources from cloud storages or from a data lake, modeling their schemas, establishing relationships, and exporting the model to an analytics platform (such as *Power BI* or *Tableau*). All the connection details to the different storage platforms are hidden by abstraction and users may start generating insights based on the semantic model created by them.

A similar approach is followed by the *Dremio Lakehouse Platform*.[9] Just like the *Semantic Layer* by AtScale, *Dremio*'s semantic layer manifests into a hierarchical structure that exposes business representations of an organization's data assets to enable exploration.

3.2. Semantic modeling for data integration

Semantic modeling targets at providing a semantic description of data sources and is not restricted to data lakes, but can be applied to data management in general. Nevertheless, methods for enriching metadata are very relevant in this context, because they increase the usability of a data lake. The semantic model acts as a bridge between the data source and a semantically rich KG, which can be applied in data integration or ontology-based data access. In this section, we first focus on approaches for creating semantic models semi-automatically (Section 3.2.1 & Section 3.2.2), potentially supported by human input in a graphical user interface (GUI) (Section 3.2.3). The manual definition of semantic models is possible but is not feasible for a large number of heterogeneous data sets as expected in data lakes. Thus, automating the generation of semantic labels and models helps to reduce manual efforts in a semantic data lake system [66].

3.2.1. Automated assignment of semantic labels

The technique of bootstrapping an initial ontology from (relational) databases and directly establishing a mapping between the databases and the created concepts is present in several systems. For example, *Optique* as well as its successor *OnTop* (see Section 4.1) provide such capabilities. An early benchmark for evaluating the quality of the generated mappings in the context of ontology-based data access (OBDA, see Section 4) is presented by *RODI* [67]. Influential works include the *SemanticTyper* presented by Ramnandan et al. [40], who apply heuristic rules (a TF-IDF-based approach for textual data and the Kolmogorov–Smirnov statistical hypothesis test for numeric data) as well as the Domain-Independent Semantic Labeler (*DSL*) by Pham et al. [41], which presents a machine learning approach around similarity measures.

As mentioned previously, interested readers are referred to the survey by Liu et al. [27] for a thorough discussion of proposals for STDs up until the year 2021 including the numerous developments based on deep learning. Here, we want to include one more recent approach called *Tab2KG* ([68], 2022), which follows an entirely different approach in the sense that it can interpret previously unseen data based on arbitrary domain KGs. Interestingly, *Tab2KG* uses RML to serialize semantic labels possibly useful to perform OBDA (see Section 4) They introduce so-called semantic profiles for domain ontologies and data tables to facilitate effective semantic table interpretation. For creating a domain profile, the domain KG, which contains representative values for the data type relations in the target domain, is transformed into a feature vector containing a set of statistics, computed using all literals coming from to set of data type relations. For creating a data table profile, a feature vector of descriptive values is computed which includes data types and basic statistics (completeness, mean, standard deviation, skewness, histograms, etc.). The selection is motivated by the expected feature effectiveness for semantic table interpretation, i.e., matching the domain and data table profiles, and can be extended

[5] https://www.stardog.com/
[6] https://www.poolparty.biz/
[7] https://cambridgesemantics.com/solutions-3/asdl-2/
[8] https://www.atscale.com/
[9] https://www.dremio.com/resources/demos/dremio-semantic-layer/

to include relevant domain-specific characteristics. The profiles are generated automatically and described using the DCAT2 and the SEAS3 vocabularies to enable their reusability. Then, Tab2KG uses the domain and data table profiles to perform the semantic type detection using a novel one-shot learning approach. In contrast to domain-specific approaches such as *DoDuo*[27,33], *Tab2KG* uses a Siamese network that generalizes towards unseen data type relations by inducing a metric that represents the domain-independent similarity between two input feature vectors (e.g., between an unknown and a known sample). The similarity between a column and a data type relation is predicted based on the experience of the similarity of other profiles learned earlier. Given a set of candidate column mappings with their respective similarity scores for each column in the input data table, they map each table column to a data type relation in a greedy manner.

3.2.2. Automated semantic model generation

While the previous approaches focus on semantic labels solely, there also exist some that aim to construct entire semantic models (some including semantic label generation as well), and the field has attracted increasing interest lately.

Vu et al. [69] present *PGM-SM* which uses probabilistic graphical models to create semantic models. From previously generated semantic models and the detected semantic labels, a conditional random field is trained to distinguish between good and bad semantic models. The graph is then used to identify possible semantic models and the best k out of them are selected using a scoring function based on probabilistic graphical models. In follow-up work, [70] they recognize that despite being flexible in choosing a target ontology, the approach requires users to label enough data sources before the systems can achieve good performance and this issue is more profound the larger the target ontology is. For this reason, they shift the supervised problem towards utilizing large-scale KGs, such as DBPedia and WikiData. For this, they present a novel probabilistic approach for automatically building semantic descriptions of Wikipedia tables leveraging hyperlinks and existing knowledge in Wikidata to construct a graph of possible relationships in the table and its context. To assess the effectiveness of the method, it is evaluated on a data set from the SemTab2020 challenge as well as another data set with 250 Wikipedia tables with their semantic descriptions built using the Wikidata ontology.

Further notable works related to the SemTab2020 challenge are MantisTable [71], bbw [72], and MTab4WikiData [73] each of them aiming at the automatic generation of semantic models for data from Wikipedia or Wikidata.

In the spirit of *CoreKG*, Feng et al. [74] present *ASMaaS* (Automatic Semantic Modeling as a Service), which is an external service for data lake vendors and users to generate a semantic model for their data sets. The paper describes a service-oriented architecture including annotating training data, training the machine learning models, and predicting an accurate semantic model for new data sources.

Futia et al. [75] present an approach called *SeMi* (SEmantic Modeling machIne) based on graph neural networks (GNNs), that was inspired by the work of Taheryan et al. [76–78], Knoblock et al. [79] (who are also co-founders of *Karma*) as well as the *SemanticTyper* by Ramnandan covering the process of semantic modeling. They adopt the exploitation of linked data repositories as background knowledge similar to Taheriyan et al. They replace the manual extraction of compound features (e.g., complex graph patterns to represent semantic relationships of different lengths) with a graph neural network that automatically learns latent features for entities and properties, encoding them in a vector space and exploiting the local neighborhood structures within the linked data graph. In their processing pipeline, the Semantic Model Builder generates an initial semantic model. The proposed Steiner tree within the graph includes the shortest path to connect semantic type classes. However, it does not necessarily express the correct semantic description of the target source. For this reason, a refinement process is required to identify a more accurate semantic model. The semantic model refinement (notice, that here an automatic process is applied as opposed to the one in *PLASMA*, cf. Fig. 3) requires preparing data sets as input of the deep learning model. The graph neural network's main goal is to reconstruct the linked data edges using the latent representation of entities and properties.

Burgdorf et al. [37] present *VC-SLAM*, a corpus that allows the evaluation and comparison of semantic labeling and modeling approaches across different methodologies. It contains 101 data sets composed of labels, data, and metadata, as well as corresponding semantic labels and a semantic model. The semantic labels and the semantic model are manually refined by human experts using an ontology that was explicitly built for the corpus. While the first two directions in the semantic labeling process (see Section 2), i.e., data- & schema-driven, have been addressed by the majority of the works presented so far, the third direction has attracted only little attention so far. Metadata-based semantic labeling regards all available additional pieces of information on a data set that might contribute to the semantic labeling of the data. For example, structured data like CKAN standards and any additional metadata within a database, like comments or textual data documentation stored in documentation tools or Wikis could be used to enhance the semantic labeling process. They have tried to evaluate *SeMi* against their datasets, but were unable because they have created a novel domain ontology that is not reasonably big enough to train the GNN with background data. For this reason, they include additional exploitable textual data documentation.

Burgdorf et al. present also *DocSemMap* [80] a novel approach that utilizes textual data documentations of data sets as an additional source for the creation of semantic mappings. It utilizes Natural Language Processing (NLP) techniques jointly with structured data to perform semantic modeling. They show that especially when compared to data-driven methods, there are different cases where *DocSemMap* can map concepts that would not otherwise be identified. Hence there is great potential for performing semantic mapping based on textual data documentation.

In [81], Ramirez et al. present Relational Natural Language Inference (*RLNI*) aiming to provide explainable data exploration on data lakes. They address the fact that many frameworks tend to build on similarity metrics that stop short of providing a clear explanation as to how an identified data set relates to a provided target. To tackle this problem they focus on based on Natural Language Inference (NLI), which has a limited set of relationships (i.e., equivalence, forward entailment, reverse entailment, negation, alternation, cover, and independence), and provides an unsupervised analysis as well as a supervised alternative, that requires training data both based on pre-trained language models. The approach fails to incorporate the semantic expressivity inherent to full-fledged ontologies but rather provides semantics and explainability to the problem of exploration in data lakes which is recognized as insufficient in comparable works such as *Aurum* [32,82].

Haller et al. [83,84] study the whole problem from a different angle. In particular, they analyze SQL queries written by data analysts, who already understand the semantic relationships of the heterogeneous data sources. Their prototype called *Pharos* connects to an existing SQL session of any program that has a JDBC interface, and records and analyzes all queries in the background. It extracts knowledge fragments from SQL queries and represents them in an RDF-based KG for which they have designed a new ontology. Examples include foreign/primary key constructs extracted from JOIN queries or log entries of the form "attribute salary has been cast to the data type long" from which it can be concluded that the salary column can be summed.

3.2.3. Semantic modeling systems

The systems mentioned in Section 3.1.1 to Section 3.1.3 focus on *semantic labeling*, i.e., enable the annotation of data sets with elements from internal and external ontologies. *Semantic modeling* is the next step in semantic enrichment: the associated ontology elements are enriched

with further semantics to reflect the relationships represented in the data sets (see Fig. 1).

ESKAPE [85] is a semantic data platform handling the full data management process from ingestion to extraction supporting heterogeneous data sources. It introduces an intuitive interface where users can create semantic models. In [2], the authors report the integration of *ESKAPE* into a data lake in a production setting. Instead of just dumping raw data into the data lake directly, they use *ESKAPE* during data ingestion to enable the semantic annotation of data sets to perform semantic data integration. To combine the semantic models, *ESKAPE* allows the integration of external KGs, such as WordNet, BabelNet, or domain-specific ontologies. Elements of the KGs can be reused during the modeling process and therefore serve as anchors when relating different distinct semantic models and data sets.

Karma [86] is a data integration tool that enables users to ingest data as well as a vocabulary defined as an OWL ontology to create a corresponding semantic model. *Karma* supports multiple data formats and proposes an initial semantic model which the users can edit using a GUI. *Karma* is an important pioneering work being one of the first to introduce key technologies such as conditional random fields (CRF) and an algorithm to solve a Steiner tree problem for learning of semantic labels and to discover relationships among the schema elements of a source. Once the semantic modeling based on the ontology is completed, Karma integrates data sets based on the defined semantic model. The resulting semantic models can be exported as a RDF file. In contrast to *ESKAPE*, *Karma* provides initial semantic labeling suggestions, but it also requires users to have in-depth knowledge about technical details, e.g., the RDF format. Furthermore, it focuses on automatically creating semantic models based on a fixed ontology per use case where the ontology does not evolve over time. *ESKAPE* claims to be easier to use because it hides the technical details; however, there is no implementation available to compare the usability.

Recently, the open-source platform *PLASMA* [18] has emerged from *ESKAPE* as well as several other notable works, described in more detail in the next section. Paulus et al. introduce a phase called *semantic refinement* (see Fig. 3) in which the human is responsible for improving the quality of the semantic model by validating, correcting, selecting, or exchanging concepts. *PLASMA* comes with a semantic concept recommendation framework to suggest semantic concepts extracted from public KGs as well as any proprietary source (e.g., a company's KG). The semantic models can be created, deleted, searched, and exported as RDF in Turtle syntax. A central idea of the framework are continuously evolving ontologies. As sources need to be added or updated, the created semantic model must be extended or adapted to reflect these changes. To address this issue, Pomp et al. [87] propose an approach featuring an evolutionary KG that consists of an internally growing universal KG and data source-specific mappings. A user, who creates a semantic label, is enabled to define and use concepts and relationships in their semantic models previously unknown to the KG. This is, for instance, the case if the concept the user wants to use is missing in the current conceptualizations. This novel knowledge must be integrated into the ontology of the universal KG. For this, Pomp et al. developed a strategy for the controlled and evolutionary development of the KG which evaluates the relationships and concepts that a user selects or introduces, and validates if the user introduces contradictions to the current KG. The framework identifies additional relationships between concepts that improve the density of the underlying ontology and contains validation logic to eliminate inconsistencies.

The dataspace management system On-demand Data INtegration (*ODIN*) [51] allows data integration by virtually querying heterogeneous data sources grounded on KGs. *ODIN* automatically extracts the schemata from (semi-)structured data sources, translates them into a canonical data model, aligns their schemata, and generates target-specific metadata from them. In the spirit of semantic refinement, it relies on a supported user feedback process for incrementally merging the semantics into a dataspace-wide ontology.[10] *ODIN* supports an advanced visual query mechanism by relying on *WebVOWl*,[11] which is an open-source application for interactive visualization of ontologies. This ontology-mediated query interface allows selecting nodes of interest from any available semantic model by marking them graphically to generate a SPARQL query. Data sources associated with the marked nodes represent the connection point to the data sets; wrappers encode the query to extract their data and expose a first-normal form relation of their schemata.

3.3. Comparison & discussion

First, it should be noted that all of the mentioned approaches focus on relational/tabular or at least (semi-) structured data (including JSON and XML) only. The heterogeneity inherent to Big Data applications is still far from being covered for automatic approaches.

Table 1 compares various proposals for data lake systems and metadata models.

By assessing the criteria we conclude that *CoreKG* comes with relevant functionality to implement SDM in practice. However, as previously described *CoreKG* is meant to be used in conjunction with the storing and processing capabilities of a full-fledged data lake system, because it does not come with any. In contrast, the *Enterprise KG Platform by Stardog* appears to be highly mature. However, its lack of open-source availability currently renders it irrelevant to academia.

Semantic Type Detection has gained research momentum lately, but the field is far from a truly mature solution. The latest solutions based on deep learning achieve impressive accuracies when evaluated on their specific test data sets, but because any labeled data set for performing supervised learning (such as *WikiTables*[12] or *GitTables*[13]) can only contain a limited amount of semantic types (i.e. labels), its usefulness in practice becomes very limited. They ultimately suffer from a closed-world assumption as in the case of *DoDuo* [33]: their final model can distinguish only between 255 different types. To sensitize the model for a different domain ontology (other than WikiData), it is required to train the model on a manually labeled data set containing corresponding semantic types. Furthermore, because they use a language model, the detection performs poorly on numeric data so far and most approaches are limited in dealing with high-dimensional and large data sets. Taking up *DoDuo* again, it has been trained on tables with an explicit maximum of 10 columns and 32 tokens in sequence length. In contrast, the embedding approach in *DAGOBAH* or the heuristic strategies by *MTab* and *JenTab* [88–90] may possibly predict arbitrary semantic types, because here semantic types are extracted via look-ups, that are not limited to specific targets. However, both strategies evaluate their algorithms on excerpts of cross-domain KGs such as WikiData and DBpedia and are not specifically designed to work with a specialized conceptualization, such as a domain ontology. This is particularly important for the applicability to data lakes, as they are often utilized in a specific domain (e.g., smart manufacturing [14,91] or the energy sector [92]) and specialized knowledge is needed for in-depth data understanding.

In terms of automated modeling, *SeMi*'s approach relying on KG embeddings computed with GNNs is an advanced solution, but its semantic type detection, the first and most crucial step in the pipeline, is not generic but tailored towards the specific data sets [75].

In Table 2 we review complete modeling platforms, that include methods for semantic labeling and/or modeling, as well as a graphical user interface to allow the user the adaptation of the created artifacts.

[10] A demonstration of the system is available at https://www.essi.upc.edu/~snadal/odin.html.

[11] http://vowl.visualdataweb.org/webvowl.html

[12] http://websail-fe.cs.northwestern.edu/TabEL/

[13] https://gittables.github.io/

In terms of GUI-based manual modeling *PLASMA* displays a mature solution, proven to be applicable in production with many built-in features that can be tested publicly.[14]

4. Ontology-based data access & integration

With semantic labels and semantic models, a semantic description of the data sources is available; it is hence reasonable to use these semantics for data integration and querying. This is the topic of *ontology-based data access* (OBDA) which has been developed since the 2000s [93] to facilitate access to various types of data sources. OBDA uses an ontology as the common data model. Calvanese et al. [94] introduced the term ontology-based data integration (OBDI) to emphasize that data from multiple sources is accessed *and* integrated. However, as the term OBDA is used more commonly than OBDI, we decided to use this term in this survey, also when multiple sources are accessed through an OBDA system. In OBDA, a set of existing data sources form the data layer. The goal is to build a service on top of this, aiming at presenting a conceptual view of data to the clients. The ontology describes the domain of interest and is usually developed independently from the data layer.

A challenge in this scenario is the combination of different abstraction levels and formalisms. Whereas ontologies model the conceptual level, use open-world assumption, and are based on description logics, the data sources are described by schemas on the logical level which focus on the structure of the data and use close-world assumption. By posing the query to the ontology, the primary advantage of OBDA is that users can query various data sources without knowing how the data is organized and where it is stored.

Fathy et al. [95] identify two main approaches for applying OBDA: (1) **materialization**, meaning all source data sets are converted into a common data format (e.g., RDF or relational tables) and stored in one data repository, or (2) **on-demand translation** (or rewriting) of queries (usually expressed in SPARQL) into the required query language of the data sources (see Fig. 5). Thus, the main difference is whether to translate the data or the query. In this survey, we emphasize the applicability to Big Data; hence, scalability is a strong requirement. Materialization approaches that involve the transformation of huge data sets into a common format can be challenging and expensive. Albeit the fact that it is significantly more complex because it requires profound theoretical consideration, only query translation approaches are feasible in data lakes. This defines the scope of this section; we only consider methods that perform query processing based on query rewriting and disregard any approaches that perform materialization. To name a few examples: *Quarry* [96], RDFizer[15] [97] or *SPARQL Anything* [98] solve semantic data integration by transforming the data into a common format (usually RDF) and then executing the query on the materialized data.

Query rewriting is part of the larger domain of data federation [99] which addresses the problem of uniformly accessing multiple, possibly heterogeneous data sources, by mapping them into a unified schema. The common data model used for querying can be also expressed in other schema languages. Earlier approaches for data integration used the relational model and translated the data sources into a common relational schema [100]. Query rewriting in this relational setting has been well studied; we consider the more complex scenario of query translation with ontologies over heterogeneous sources. Query translation [101] is hence strongly related and essential for any effective OBDA framework to deal with the variety of heterogeneous data sources in a data lake. In OBDA, on-demand query translation exposes the content of data sources as RDF triples, using classes and predicates from a KG. The RDF triples are not materialized, they are part of a virtual knowledge graph (VKG), which means that data remains in the data sources instead of being stored in some common database. A VKG system has the following components (see Fig. 5): (a) queries that describe user information needs, (b) an ontology with classes and properties, (c) mappings to the data sources, and (d) a collection of data sources. The W3C published recommendations for languages for components (a)–(c): SPARQL, OWL 2, and R2RML,[16] respectively.

Mapping creation and management is probably the most complicated OBDA design-time task, as the mapping specifies the semantics of the data sources in terms of the ontology and bridges the typically large conceptual gap between the source schema and the ontology. By applying knowledge representation and automated reasoning techniques, an OBDA system uses ontologies and mappings to reformulate the SPARQL queries into source-specific queries. A formal framework is presented by [8], including query answering, mapping management and analysis, and extensions of the classical OBDA framework. Here, we omit the theoretical details of query rewriting and focus on the practical application for data integration in the Big Data regime.

A comprehensive survey on query translation methods by Mami et al. is given in [101], including a classification, a map for different language translation paths, and a historical timeline. While computer scientists have been looking for the holy grail of data representation for decades (logic, ER/relational, XML, graph, NoSQL), their work reinforces the view that no single data representation scheme fits all use cases. Mami et al. further identify several interesting shortcomings: First, they recognize that many translation methods fail to support the more sophisticated operations such as different join types and temporal functions in SQL, or blank nodes, grouping, and binding in SPARQL. Then, there are many well-founded and defined query translation frameworks, from the query translation process to the various optimization strategies; however, they would hardly represent real-world queries, but mostly simple queries. Use-case-driven translation methods would be more helpful to reveal useful query patterns and to evaluate the translation methods and optimizations on real-world data. They state that there is a wide variety in the evaluation frameworks used by each of the query translation methods and a need for a unique standardized benchmark specialized in evaluating and assessing query translation aspects. In their final remarks, they discuss candidates for a 'universal' query language. Based on their findings exploring various query translation methods, they see SQL and SPARQL as the most suitable languages to act as a 'universal' language for realizing heterogeneous data integration.

4.1. OBDA over relational data

Some of the earliest systems performing OBDA include *Mastro* (2011, [102]), *UltraWrap* (2013, [103]) and *MorphDB* (2014, [104]). *Optique* (2013 [105]) was the first successful OBDA system designed for Big Data scenarios. Its underlying architecture mainly uses *Ontop* (see below) to manage data access using R2RML mappings, and Exareme [106], a research prototype that acts as a back-end query execution component handling large-scale data processing tasks. A major drawback of these early systems is the lack of horizontal scalability.

The *Ontop* framework for OBDA [107] is likely to be one of the most mature open-source[17] OBDA systems, which has come a long way since its beginnings in 2009 [107,108]. It is the result of an active research and development community and has been adopted in many academic and industrial projects [4,109,110]. Because *Ontop* moved towards supporting the W3C recommendations for SPARQL and R2RML, new challenges emerged, which required the development of a core data structure called intermediate query (IQ), an algebra-based data structure that unifies both SPARQL and relational algebra. IQ is a uniform

[14] http://plasma.uni-wuppertal.de/modelings
[15] https://github.com/SDM-TIB/SDM-RDFizer
[16] https://www.w3.org/TR/r2rml/
[17] https://github.com/ontop/ontop

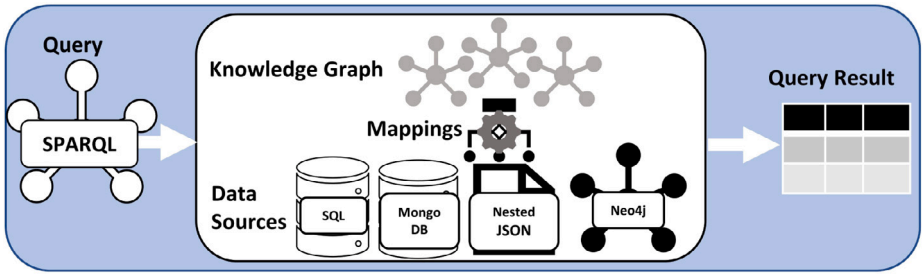

Fig. 5. Illustration of Ontology-Based Data Access.

representation both for SPARQL queries and for SQL queries from the mapping. When the query transformation (rewriting and unfolding) is complete, the IQ expression is converted into SQL, and executed by the underlying relational database management system (DBMS). *OnTop* aims to shift almost all query processing to the DBMS and performs only the topmost projection, which typically transforms database values into RDF terms. Sophisticated optimization techniques such as redundant join elimination, using primary and foreign keys, efficient duplicate elimination [111], and pushing down joins to the data-level [112] as well as exploiting integrity constraints to enhance query completeness and performance [113] have been extensively studied during the past decade to make sure that queries can be efficiently processed by the underlying DBMS.

With a recent extension called *Ontop4theWeb* [114], the framework can now be evaluated on the fly against web data using SPARQL without materializing it into a DBMS first. *Ontop4theWeb* implements an approach for posing SPARQL queries on top of non-RDF Web data on the fly. To achieve this, virtual table operators are embedded in the SQL queries that are included in R2RML mappings. They specify which part and source of Web data will be fetched and how they will be mapped to virtual RDF terms. The specification of these mappings creates the usual overhead, but they need to be specified only once unless the schema changes. Although no explicit materialization of data is performed and the responses of web-native and NoSQL JSON APIs can be obtained, *Ontop4theWeb* still relies on the relational data model and a relational database system in the backend.

Ontopic Studio[18] is a commercial spin-off of *Ontop* that provides a GUI to ease the creation and search of mappings via a mapping editor. The *Ontop* framework has recently attracted interest for an application in data lakes: Schwade et al. [115] show a successful integration into a data lake by adding a semantic layer based on the *Ontop* framework.

The *Chimera Suite* [116] extends the native *OnTop* systems by two components: (1) Ontop$_{Spark}$, an extension that enables *OnTop* to perform OBDA on relational data using Apache Spark effectively translating *SPARQL* queries into *SparkSQL* and (2) a JDBC endpoint to connect *Spark Thrift Server* with *OnTop* that can direct the *SparkSQL* queries to an HDFS filesystem. Belcao et al. point out the current gap between Big Data technologies and the semantic world. On the one hand, knowledge engineers work with OWL and RDF to create knowledge graphs. On the other hand, data engineers define ETL processes to provide the source data in a form that can be consumed by data scientists. Then, data scientists need to combine these two perspectives to provide interpretable analytics results. To tackle this problem, Belcao et al. introduce a comprehensive pipeline, where data scientists write SPARQL queries and send them to Jena Fuseki in a Jupyter Notebook using the PySPARQL library to get a semantically enriched response. Using the mappings and the ontology, the query is re-written into SQL. Ontop$_{Spark}$ accesses the data stored as Parquet files in HDFS using Apache Spark. The SPARQL response returned from Ontop$_{Spark}$ is further enriched by Jena Fuseki using the knowledge graph, and the final results are sent back to the Jupyter Notebook.

Belcao et al. further describe in [116] a real-world deployment at Italy's largest public company for industrial research in the energy sector [3]. In this scenario, they developed a KG containing around 7 million triples representing Milan's metropolitan area storing the network topology and all the information of electrical equipment. The *Chimera Suite* enables the data science team to make analytical predictions by querying both the KG and the Big Data repository. The knowledge engineers can access the graph and retrieve the results of the data scientists' analysis.

Ontario [117] is a query processing engine over heterogeneous data sources in a data lake utilizing VKGs. *Ontario* translates queries from a global querying mechanism, e.g., SPARQL, to the underlying native query mechanism of the data sources. Therefore, the system transforms raw data to RDF on demand by applying mapping rules based on R2RML. *Ontario* utilizes custom extensions of the RDF and SPARQL standards, where compliance with W3C standards is unclear.

Obi-Wan [118] is built on top of the *Tatooine* [119] mediator for heterogeneous sources and *Graal*, a toolkit for query answering in knowledge bases [120]. The approach relies on mappings that are Global-Local-As-View (GLAV) as opposed to Global-As-View (GAV), which are used, for example, in *Ontop*, *MorphDB* and *Mastro*. GLAV mappings are considered to be more flexible in scenarios with evolving data sources [100]. In GAV mappings, a change in a data source might require a change in the global schema and all source mappings. With GLAV mappings, the sources are described independently of each other concerning the global schema.

This was also recognized by the work of Nadal et al. [121] who propose a framework for data integration entirely based on graphs utilizing GLAV mappings. A prototype of the rewriting algorithms is incorporated into *ODIN* (see Section 3.2). The proposed query language is based on the coverings of a graph representing the global schema and does not require users to define join conditions. The query is visually represented as a graph. Required join operations on the data are generated by the rewriting algorithm. Encoding all required metadata (i.e., global schema, source descriptions, mappings, and queries) as graphs simplifies the interoperability among them and allows rewriting algorithms to efficiently identify the relevant sources.

PolyWeb [122] relies on R2RML and its proposed successor RML.[19] *PolyWeb* performs a predicate-based source selection where a set of relevant data sources on the Web (RDF & CSV) for a given query are discovered by matching predicates used in a basic graph pattern and the input data sources. A SPARQL query provided as input is translated into a native query of the underlying system (CSV files, relational databases, and RDF triple stores are considered) and executed on this system.

[18] https://ontopic.ai/

[19] https://rml.io/specs/rml/

4.2. OBDA over non-relational data

There are four main groups of NoSQL Database Management Systems (DBMS) [95]: document-oriented, column-oriented, graph-oriented, and key–value stores. Each of these DBMSs relies on different data models and query engines, which complicates the task of creating a uniform OBDA system in this case. Because of increasing popularity and the lack of unified query languages and data models for these systems, there is growing interest in applying OBDA in this context. Systems based on VKGs promise to provide the desired unified semantic view of data sets. Early papers include [123,124] and different languages have been proposed to extend the R2RML mapping language.

RML [125] is such a proposed extension to deal with heterogeneous data sources. There are proposals to extend RML with functions and operations described uniformly, unambiguously, and independently of the technology used for implementation. Examples of those include the Function Ontology (FnO) [126] and FunUL [127]. While RML extends R2RML to schema transformations, the combination of RML with FnO extends R2RML concerning data transformations. RML is widely spread and implemented in existing systems, and many proposals exist that aim to ease its use. YARRRML [128] simplifies the use of both R2RML and RML by a human-readable text-based representation for mapping rules expressed in YAML,[20] a widely used human-friendly data serialization language. The RML Editor [129], RMLx [130] Map-On (only for R2RML [131]) and MapVOWL [132] provide visual support to help users in generating Linked Data from raw data by defining RML mappings. In [133] the authors conduct a usability experiment on three different languages: ShExML [133], YARRRML and SPARQL-Generate [134]. RML was left out, because of too high verbosity. Their prototype translates files (JSON, XML) into RDF using one of the mappings languages and examines the usability via a questionnaire for first-time users, which are have only basic skills in programming and linked data. The results show that ShExML users tend to perform better than those of YARRRML and SPARQL-Generate. With SDM-RDFizer and RocketRML [97,135], there exist further concrete solutions that built upon RML. However, these approaches, as well as some of the ones mentioned before, follow the materialized approach, i.e., converting all data into a huge RDF file as opposed to on-demand query rewriting.

xR2RML is a language for mapping various types of databases to RDF which enables the translation of a broad scope of data sources via query re-writing [136]. To achieve this goal, they propose a two-phase approach. First, they define an abstract query language derived from SPARQL. Utilizing the xR2RML mapping language and leveraging R2RML-based SPARQL-to-SQL works, they introduce a generic method to translate a SPARQL graph pattern into their abstract query language. In the second phase, the abstract query is translated into the query language of a target database. For demonstration purposes, they apply the approach to MongoDB but acknowledge that adopting a different NoSQL DBMS will still be challenging. This is mainly because these systems are generally optimized for fast storage and retrieval of vast collections of data, rather than on expressive query languages. The prototype implementation is open-source[21] including the SPARQL-to-MongoDB translator and connectors for the MySQL and PostgreSQL relational databases.

Delva et al. [137] identify certain shortcomings of ShExML, and xR2RML and combine concepts of both to introduce RMLFields. It extends RML with a nested iteration model empowering it to write nested loops over input data. In another approach [138] called *OntoMongo* the authors seek to delegate query execution to a NoSQL source engine. They rely on an object-oriented intermediate representation, where they map from the ontology vocabulary to this object-oriented, conceptual layer. The aim is to simplify the mapping specification and make it independent of the underlying source database.

A team around the creators of *Ontop* has been working on generalizing the OBDA concepts to non-relational data models. They have carried out experiments on MongoDB to extend the functionality of the system [139]. They explain a two-step rewriting process of SPARQL queries into the MongoDB aggregate query language, where they represent the results of native queries as relations to provide a uniform view of non-relational queries. The idea is to view a collection of MongoDB documents in the nested relational model.

A similar implementation is reported in [140] aiming to fill the gap between NoSQL and the Semantic Web, to enable access to such databases and integration of non-relational data sources. They present the *Ontop-CB* project which implements the query translation method based on the *Ontop* system as a backbone, allowing to query Couchbase, a NoSQL document store. The key components are an OWL ontology, an access interface, mappings, a NoSQL database, a SPARQL to NoSQL query adjustment, and a JSON export. The approach exploits *Ontop* answers to SPARQL queries by rewriting them into SQL queries and delegating their execution to the database. To do so, they also establish an object-oriented intermediate layer between the OWL ontology and the Couchbase data source.

The authors of [141] report the implementation of OBDA for a Neo4j graph database via two phases. First, the xR2RML mapping language is used to connect the RDF data model to the LPG data model. In the LPG data model, edges may have multiple data properties, referred to as 'Composite Edge'. Then, relational graph algebra is used to translate a given SPARQL query into the Cypher query language native to Neo4j.

In line with the use of OBDA in NoSQL, the problem of ontology-mediated query answering over key–value stores is studied in [142]. The authors create a rule-based language in which keys are used as unary predicates and rules are applied at the record stage (a record is a set of key–value pairs).

Chimera's main competitor is the Semantic ANalytics StAck (*SANSA*) [19,143,144], that comes with *Squerall*. *Squerall* is a scalable OBDA engine that relies on the RML standard and allows querying several databases simultaneously. *Squerall* uses wrappers to query heterogeneous data sources directly in their original form. In the paper, they evaluate and compare five different data sources (Cassandra, MySQL, MongoDB, Parquet, and CSV) and run queries involving multiple joins between the different sources. They compare the *Presto* and *Apache Spark* processing engines for making *SPARQL-to-SQL* conversions and find out that *Presto* works faster in their specific evaluation. *Squerall* addresses the variety challenge of Big Data in that it can conveniently be extended to embrace new data sources. This is the most immediate advantage over *Chimera* and *Ontop*, because those usually require a single DBMS (or *Hive* store) in the backend. However, *Squerall* does not comply with many W3C Recommendations. For example, basic SPARQL graph patterns OPTIONAL and UNION were not yet supported at the time of the first publication in 2019. Similarly, it is unclear how generic the framework truly is from a theoretical perspective beyond the evaluation scenario given in the paper. However, the work is a crucial step in the vision of implementing a polystore system that can query data in an on-the-fly manner, i.e., no prior transformation.

4.3. Discussion

R2RML has been proven to be a useful mapping language for relational sources. Simply extending the R2RML standard to support other types of sources does not necessarily carry on all its features [145]. For example, select conditions and transformation functions are supported by R2RML implicitly relying on the expressivity of SQL, but this cannot be applied for querying XML or JSON documents. In [146], the authors argue first, typical mapping languages are designed to work with a specific data format, and second, they are designed in a format to be parsed by machines and they do not take into account human readability and compactness. YARRRML is a contribution in that direction. Because

[20] https://yaml.org/spec/
[21] https://github.com/frmichel/morph-xr2rml/

Table 3
Overview of OBDA techniques sorted by date of publication.

	Year	Data model	Query languages	Mapping model
Maestro	2011	Relational	SQL	*DL-Lite*
UltraWrap	2013	Relational	SQL	RDB2RDF
Optique	2013	Relational	SQL	RDB2RDF
MorphDB	2014	Relational	SQL	R2RML
OnTop	2014	Relational	SQL	R2RML
Mugnier et al. [142]	2017	SQL & Key-Value	SQL, XPath, JSONPath, MongoDB	NO-RL
OntoMongo	2017	Relational & Document	SQL & MongoDB	Object-relational & Object-Document
PolyWeb	2019	Relational	SQL	R2RML & RML
OnTop over MongoDB	2019	Relational & Document	SQL & MongoDB	JSON-to-RDF & SQL-to-RDF
Ontario	2019	RDF & Relational	SQL	RDF-MT
Squerral (*SANSA*)	2019	Relational & NoSQL	Spark- & Presto-SQL	RML+FNO
Fathy et al. [141]	2019	labeled property graph	Cypher	xR2RML
Obi-Wan	2020	Relational & Document	SQL & MongoDB	(G)LAV view-based query rewriting
OnTop4theWeb	2021	REST (CSV, JSON, XML)	SPARQL	R2RML
Chimera	2021	Relational (Hive)	SparkSQL	R2RML
OntoCB	2021	Document	Couchbase (N1QL)	Object-oriented

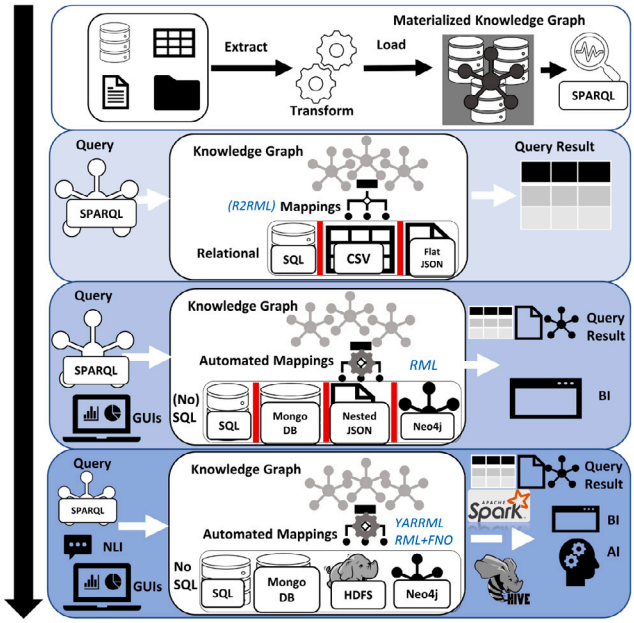

Fig. 6. Timeline of OBDA techniques.

these languages are not necessarily interoperable, and many of them support a specific engine only, the next generation of OBDA systems would have to consider how to address translation mapping languages, in addition to the query re-writing techniques that have been widely addressed in the SOTA of OBDA so far (see Fig. 6).

Squerall extracts data, transfers it to an efficient format, and queries it in a scalable manner using *Apache Spark*. In contrast, *Ontop* predates the publication of the *Chimera* and therefore does not utilize any Big Data technology. However, the comparison reported states that *Ontop* is still competitive for the majority of the evaluated query types (12 in total), except for such queries that involve aggregating, where *Squerall* is more efficient. This might come from the fact that *OnTop* has more sophisticated query optimization techniques, while *Squerall* relies solely on its processing engine. Further, *Squerall* does not support all query types.

Based on the existing literature we identify a historical development (see Table 3) of OBDA techniques which is separated into four different periods (see Fig. 6): First, there are proposals following the materialized approach, i.e., transforming all data sources into a huge RDF graph to be queried using SPARQL. Although the approach might suggest a uniform query method for heterogeneous data sources, the question of data integration remains unanswered as data is only transformed into a common format, but it is unclear how they relate semantically to each other. The approach also has the previously described deficiencies about the liveliness of data and the overhead that comes with the initial transformation into RDF.

These weak points were then addressed by the first platforms to perform OBDA over relational databases via query re-writing, which leaves the data in their respective DBMS retrieving it through the native via query optimization techniques. Although these solutions display impressive progress, they are mostly limited to supporting only a single relational DBMS at once (indicated by a red bar). The process of creating mappings is also mostly manual and hence tedious, error-prone, and tailored towards a single static ontology.

The third period bridges the gap to access NoSQL databases and more flexible data models such as nested structures, graphs, or REST APIs and also expands the query result space to NoSQL data models. In this period many 'single-purpose' solutions arise that extend support to a single NoSQL DBMS or semi-structured files like JSON and XML, but no solution exists that is flexible enough to query arbitrary databases. In this period we also observe the first real-world applications that built GUIs for these methods to perform business intelligence effectively, e.g., [4]. In parallel, the development of algorithms for automated generation of mappings gained increasing momentum, some of which have been discussed in Section 3.2. Early proposals with a focus on OBDA include the already mentioned *BootOX* complemented by *MIROR* [147], both promoting the automatic generation of W3C R2RML from RDBMSs. They are followed by Hazber et al. [148], Heyvaert et al. [149], *AutoMap4OBDA* [150], *MILAN* [151] and Iglesias-Molina et al. [152] (*Mapeathor*[22]) each presenting different algorithms to improve the mapping quality, decreasing the task complexity, and hence the user effort for creating R2RML mappings.

Finally, the most recent approaches add efficient, scalable, and distributed processing engines like *Apache Spark* or *Hive* and storages like *HDFS*. The *Chimera Suite* supports all W3C recommendations for the R2RML mapping language to translate SPARQL to SQL and bridges an important gap between semantic and Big Data technologies. However, at the same time, it is limited to *Hive* and the relational data model. Furthermore, the traditional GAV-based mappings have the drawback that one needs to revisit them if schema changes happen at the source level. For example, the *Ontop* docker container[23] needs to be restarted every time changes are made to the mapping definitions or the data sources. This problem is partly lifted by utilizing GLAV mappings, but they have not gained widespread attention yet and scalable open-source proposals are to be presented. On the other side *Squerall* supports arbitrary NoSQL databases while also utilizing scalable processing engines,

[22] https://morph.oeg.fi.upm.es/tool/mapeathor/swagger/$#$/default/post_tool_mapeathor_api_

[23] https://github.com/chimera-suite/use-case

but does not support the sophisticated SPARQL queries.[24] Furthermore, we observe the emergence of data-driven methods, in particular scaling usability of Machine Learning (ML) analytics, which requires managing a large variety of heterogeneous data with a unified mechanism, to save time on developing new ML solutions, and to reuse already developed ones. For this, companies commit to the development of long-term common semantic frameworks for industrial data management and ML analysis [153].

5. Applications

The closer collaboration between human-machine and machine-machine systems has revolutionized the current industrial landscape, leading to Industry 4.0 (also known as Industrial Internet-of-Things, IIoT) [154]. The key drivers of IIoT include the continuous addition of new types of devices with their own data models. This data explosion requires highly agile data models that enable monitoring, processing, optimizing, and analyzing data to derive insights for better decision-making. However, effective data utilization demands data integration, encompassing cleaning, de-duplication, and semantic homogenization. We proceed by presenting two hand-selected use cases where semantic data management has transcended the realm of academia and has been applied to real-world IIoT problems.

5.1. Manufacturing

The company Bosch has recognized the potential of knowledge graphs, and a series of subsequent publications highlight the company's significant advancements in the field of semantic data management and integration. Firstly, OBDA using the *SANSA Stack* as well as *OnTop* has been reported in the area of Surface Mount Technology (SMT), the process for mounting electronic components on printed-circuit boards [59,91]. This process involves several subprocesses executed by specialized machines, each generating substantial amounts of data. These data contain semantic interoperability conflicts, such as variations in object naming. To effectively explore this data, these conflicts need to be resolved by mapping them to ontology terms. The development of the SMT Ontology resulted from a collaborative effort spanning several weeks, involving a line engineer, two line managers, an SMT process expert, an SMT data manager, a project manager, two Big Data managers, and four semantic experts. Thus, adopting the VKG approach, which focuses on producing high-quality mappings, is a labor-intensive process requiring the combination of three types of knowledge initially: domain knowledge, database schema details, and understanding of the VKG approach. However, once achieved, an integrated Data View provided by the SMT Ontology enables domain experts to perform intricate product analysis tasks on previously unintegrated raw data.

The development of knowledge graphs is a major undertaking for the company, involving substantial initial effort but promising long-term solutions for various complex challenges. Firstly, it serves the primary purpose of storing knowledge and supporting material science engineers in their information search [155]. Additionally, the focus extends to data-driven methods, particularly enhancing the usability of Machine Learning (ML) analytics. This entails managing heterogeneous data from a wide array of sources through a unified mechanism, saving time in developing new ML solutions, and reusing previously developed ones [156]. Their system comprises several semantic artifacts: a core ontology that encapsulates general knowledge of the manufacturing process, domain ontologies reflecting domain-specific knowledge, and *data-to-domain ontology* mappings that link raw data attributes to domain ontology terms to enable OBDA. Several semantic modules support the management of the three fundamental building blocks: the KG generation module, a mapping reasoner/annotator, and a data integration module. The company's primary goal is to leverage ML analytics to achieve a high degree of production automation. All these components serve as prerequisites, culminating in a platform known as *SemML* (Semantic Machine Learning) [59,157], addressing three core challenges through semantically enhanced ML: (1) communication, as the workflow involves collaboration among experts from diverse areas, including data scientists, engineers, process experts, and managers with distinct backgrounds, making communication time-consuming and error-prone; (2) data integration; and (3) generalizability of ML models.

5.2. Smart city

The digital transformation encompasses not only industrial applications but also infrastructures for services of general interest, such as transportation networks and associated transport options, water and energy supply, as well as waste and wastewater disposal. These activities are commonly associated with the term *Smart City* [158]. In a Smart City, intelligent information and communication technology is employed to enhance participation, quality of life, and to create an economically, ecologically, and socially sustainable community or region.

As described in Section 3.1.1, Bagozi et al. [56] and Bianchini et al. [57,58] utilize indicators to enable personalized data lake exploration, demonstrating the model's effectiveness in the Smart City environment due to its data and user role diversity. Furthermore, Bianchini et al. [159] present a tool that implements a semantic layer over a heterogeneous ecosystem of data sources, serving as a starting point for semantics-enabled applications or data source exploration. Their pipeline involves three steps: (1) Lexical enrichment of attributes, standardizing concept names for annotation; (2) semantic annotation of data attributes; and (3) creation of semantic relationships.

Similarly, Pomp et al. [50] introduce a *Semantic Data Marketplace for Easy Data Sharing within a Smart City*, applying a predecessor of *PLASMA* (refer to Section 2) to the Smart City use case. The semantic data marketplace was employed in a project,[25] requiring teams to develop Smart City applications based on datasets provided by city departments from three different cities. Both approaches involve a Data Producer and a Data Consumer. The Data Producer, a domain expert, is responsible for uploading datasets and enriching them with semantic metadata from knowledge graphs. Using the knowledge graph and associated semantic models, data consumers can find and comprehend data by searching and reviewing semantic concepts. VC-SLAM [37] (refer to Section 3.2.2), composed of 101 datasets with corresponding semantic labels and a semantic model, has been constructed using Smart City data extracted from Open Data portals. A recent survey [160] introduces semantic Internet-of-Things technologies, which may play a crucial role in addressing essential Smart City issues, primarily interoperability challenges.

6. Challenges and conclusion

Even though we have seen various tools in which KGs play a crucial role, a comprehensive solution that provides truly scalable access to heterogeneous data-based on ontology-based mappings, including a semantic modeling component to create and maintain the mappings, is not yet available. We have illustrated this gap in Fig. 7. In the Venn diagram, we put three topics in relation: to OBDA, semantic modeling, and data lakes. Semantic Data Management may include more approaches, such as the materialization approaches neglected here. The intersection between semantic modeling and OBDA is represented for example by *ODIN*. The combination of OBDA for Big Data and data lakes is illustrated for example by the manufacturing use case [91].

[24] https://github.com/EIS-Bonn/Squerall/wiki/Squerall-Basics#3-sparql-query-interface

[25] https://youtu.be/3G9IQgK09sU

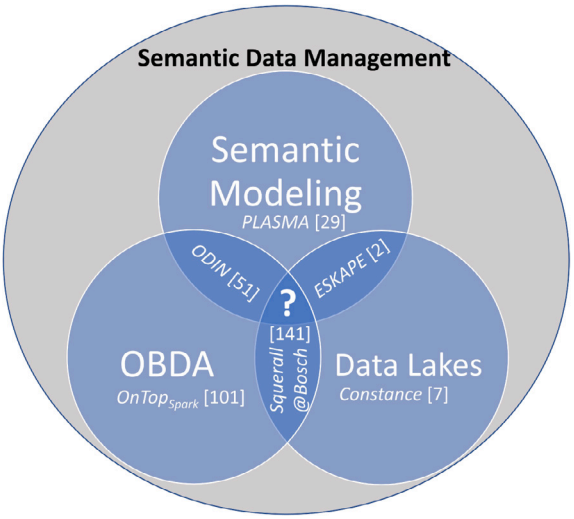

Fig. 7. Venn-Diagram to illustrate our view of the current state in the field of Semantic Data Management.

However, it involved a great amount of manual labor by experts, is not available (neither open-source nor as a commercial system) and we found no evidence that the semantic modeling techniques discussed in Section 3.2 are applied in this use case.

Semantic modeling is enabled within a data lake by *ESKAPE*. However, a custom data format named Semantic Linked Tree (SLT) is used, which is a JSON-based data format in which the semantic labels are attached to the raw data values. This transformation can be interpreted as *semantic lifting*. Due to its rather early development, the system also does not incorporate scalable data storage and processing engines inherent to most modern data lakes.

OBDA has grown from a long tradition, it has been reported in production several times, and recent proposals, such as *SANSA* and *Chimera* display meaningful progress towards semantics-based on-demand data access. However, the creation of the required mappings is a tedious, manual task for the user. There is not yet enough assistance either by good user interfaces or automatic procedures.

The field faces several challenges today:

1. *Initial overhead & usability*: Creating meaningful semantic models for a large number of heterogeneous data sets comes with a huge initial overhead in generating KGs and the corresponding connections. First, generating domain-specific conceptualizations is already time-consuming and resource-intensive, quite apart from the fact that this task is truly an entire field of its own. Even if we assume to have a versatile set of suitable KGs at hand, the creation of suitable mappings to data sources is still a time-consuming process. As presented in this survey, there are some proposals for the automatic creation of semantic models and mappings; yet, more emphasis should be put on the required human input (see *Technical abstraction*).
2. *Evaluation*: Automated generation of semantic labels and models has gained much attention lately; however, it is unclear how accurate the methods really are, beyond the scope of the test sets used in the particular publication. For semantic labeling, *SemTab* has presented various data sets and benchmarks and research is underway for more sophisticated table interpretation. For semantic models, *VC-SLAM* is the very first proposal towards standardization of a set of ontologies and data sets alongside the corresponding semantic models. The emergence of annual challenges and the publication of benchmarks will be helpful and could be further streamlined by an initiative similar to the OAEI[26] for ontology alignment.
3. *Technical interoperability*: Considering the variety of today's data landscape, one has to point out that the majority of methods for semantic labeling and modeling are based on tabular data. The field is yet far away from covering all NoSQL data models uniformly. This is also the focus of modern OBDA research. While scalable data access via ontologies can be regarded as solved to a large extent for relational data, the focus has shifted more towards incorporating different NoSQL query languages as well as supporting federated query processing against multiple databases, file systems, and other miscellaneous data sources simultaneously. Any such solution *must* be compatible with W3C standards for the Semantic Web to ensure maximal interoperability between systems.
4. *Technical abstraction*: The overall quality of a semantic model must ultimately be refined by the human operator. This human-in-the-loop process needs guidance for unfamiliar users through a strong technical abstraction, i.e., technical details must be hidden from the user. It is hard to imagine that even with very enhanced AI techniques (see below *Leverage AI*), human verification and refinement of mappings will become obsolete. Therefore, enhanced user interfaces that focus on usability also for non-technical users are required in this domain.
5. *Applicability for Big Data*: Solutions like *Squerall* and *Chimera* are promising approaches to address OBDA also in Big Data scenarios. Yet, there are many limitations in the expressiveness of queries or mappings. Furthermore, they are bound to specific versions of Big Data platforms which complicate their deployment. The maturity of such solutions could be improved if a larger community would continue the development of these prototypes, to remove the aforementioned limitations and make them applicable to a wide variety of NoSQL/Big Data systems.
6. *Leverage AI*: Since the publication of ChatGPT, it has been discussed for many human tasks whether they can be solved by Large Language Models (LLMs) in the future. This is also relevant for all tasks related to data integration as ChatGPT can describe data sources, create ontologies, and find relationships between different data sets, to name just a few examples of what can be achieved with LLMs. Given the latest development of AI techniques, we have to expect that these systems will be able to solve more complex data integration tasks in the future of which the first papers are appearing already [161]. It would be interesting to see how such general-purpose AI systems can be customized and optimized for specific data integration tasks within data lake systems at scale. On the other hand, we need to examine the processes discussed in this survey to see at which points and how an AI system based on LLMs can be integrated to improve the overall process.

To conclude, in this article we have given an overview of semantics-based methods for data management, access, and integration and related those findings to current semantic data lake proposals. Conclusively, we can state that the community faces several challenges and a gap in today's landscape between present data lake platforms, OBDA and semantic technologies for modeling the context in which heterogeneous data sets arise. We are confident that Big Data and Semantic Web technologies can benefit from each other and that more enhanced solutions for semantic data lakes will become available in the next years.

[26] http://oaei.ontologymatching.org/

CRediT authorship contribution statement

Sayed Hoseini: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Validation, Writing – original draft, Writing – review & editing. **Johannes Theissen-Lipp:** Investigation, Methodology, Validation, Writing – original draft. **Christoph Quix:** Formal analysis, Funding acquisition, Methodology, Project administration, Supervision, Validation, Writing – original draft, Writing – review & editing.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Christoph Quix reports financial support was provided by Federal Ministry of Education and Research Berlin Office.

Data availability

Data will be made available on request.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the authors used OpenAI's generative AI (*ChatGPT* v3.5) in order to improve the writing, make suggestions and for rephrasing. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the content of the publication.

Acknowledgments

This work has been sponsored by the German Federal Ministry of Education and Research in the funding program "Forschung an Fachhochschulen", project i^2DACH (grant no. 13FH557KX0).

References

[1] R. Hai, C. Quix, C. Zhou, Query rewriting for heterogeneous data lakes, in: Proc. ADBIS, in: LNCS, Vol. 11019, Springer, 2018, pp. 35–49, http://dx.doi.org/10.1007/978-3-319-98398-1_3.

[2] A. Pomp, A. Paulus, A. Kirmse, V. Kraus, T. Meisen, Applying semantics to reduce the time to analytics within complex heterogeneous infrastructures, Technologies 6 (3) (2018) 86.

[3] E. Bionda, C. Tornelli, M. Brambilla, M. Balduini, G. Mauri, D.D. Giustina, F. Garrone, E.D. Valle, The smart grid semantic platform: Synergy between IEC common information model (CIM) and big data, in: Proc. IEEE EEEIC/I&CPS Europe, 2019, pp. 1–6, http://dx.doi.org/10.1109/EEEIC.2019.8783632.

[4] E. Kharlamov, T. Mailis, G. Mehdi, C. Neuenstadt, Ö.L. Özçep, M. Roshchin, N. Solomakhina, A. Soylu, C. Svingos, S. Brandt, M. Giese, Y.E. Ioannidis, S. Lamparter, R. Möller, Y. Kotidis, A. Waaler, Semantic access to streaming and static data at Siemens, J. Web Semant. 44 (2017) 54–74, http://dx.doi.org/10.1016/j.websem.2017.02.001.

[5] M. Yahya, J.G. Breslin, M.I. Ali, Semantic web and knowledge graphs for industry 4.0, Appl. Sci. 11 (11) (2021).

[6] A. Hogan, E. Blomqvist, M. Cochez, C. d'Amato, G. de Melo, C. Gutierrez, S. Kirrane, J.E.L. Gayo, R. Navigli, S. Neumaier, A.N. Ngomo, A. Polleres, S.M. Rashid, A. Rula, L. Schmelzeisen, J.F. Sequeda, S. Staab, A. Zimmermann, Knowledge graphs, ACM Comput. Surv. 54 (4) (2022) 71:1–71:37, http://dx.doi.org/10.1145/3447772.

[7] L. Ehrlinger, W. Wöß, Towards a definition of knowledge graphs, in: Proc. SEMANTiCS & SuCCESS, in: CEUR WS, Vol. 1695, 2016.

[8] G. Xiao, D. Calvanese, R. Kontchakov, D. Lembo, A. Poggi, R. Rosati, M. Zakharyaschev, Ontology-based data access: A survey, in: Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI, ijcai.org, 2018, pp. 5511–5519.

[9] R. Hai, S. Geisler, C. Quix, Constance: An intelligent data lake system, in: Proc. ACM SIGMOD, 2016, pp. 2097–2100.

[10] M.H. Farid, A. Roatis, I.F. Ilyas, H. Hoffmann, X. Chu, CLAMS: bringing quality to data lakes, in: Proc. ACM SIGMOD, 2016, pp. 2089–2092, http://dx.doi.org/10.1145/2882903.2899391.

[11] H. Dibowski, S. Schmid, Y. Svetashova, C.A. Henson, T. Tran, Using semantic technologies to manage a data lake: Data catalog, provenance and access control, in: SSWS@ISWC, 2020, URL https://api.semanticscholar.org/CorpusID:229357020.

[12] A. Paulus, A. Burgdorf, A. Pomp, T. Meisen, Recent advances and future challenges of semantic modeling, in: Proc. 15th IEEE ICSC, IEEE, 2021, pp. 70–75, http://dx.doi.org/10.1109/ICSC50631.2021.00016.

[13] C. Bizer, T. Heath, T. Berners-Lee, Linked data, in: Semantic Services, Interoperability and Web Applications - Emerging Concepts, CRC Press, 2011, pp. 205–227, http://dx.doi.org/10.4018/978-1-60960-593-3.ch008.

[14] H. Dibowski, S. Schmid, Y. Svetashova, C. Henson, T. Tran, Using semantic technologies to manage a data lake: Data catalog, provenance and access control, in: Proc. Scalable Semantic Web Knowledge Base Systems Workshop, in: CEUR WS, Vol. 2757, 2020, pp. 65–80, URL http://ceur-ws.org/Vol-2757/SSWS2020_paper5.pdf.

[15] M.M. Cantallops, S. Sánchez-Alonso, E. García-Barriocanal, A systematic literature review on Wikidata, Data Technol. Appl. 53 (3) (2019) 250–268, http://dx.doi.org/10.1108/DTA-12-2018-0110.

[16] A. Pomp, J. Lipp, T. Meisen, You are missing a concept! enhancing ontology-based data access with evolving ontologies, in: Proc. ICSC, IEEE, 2019, pp. 98–105, http://dx.doi.org/10.1109/ICOSC.2019.8665620.

[17] C. Quix, R. Hai, Data lake, in: Encyclopedia of Big Data Technologies, Springer, 2019, http://dx.doi.org/10.1007/978-3-319-63962-8_7-1.

[18] A. Paulus, A. Burgdorf, L. Puleikis, T. Langer, A. Pomp, T. Meisen, PLASMA: platform for auxiliary semantic modeling approaches, in: Proc. ICEIS, SCITEPRESS, 2021, pp. 403–412, http://dx.doi.org/10.5220/0010499604030412.

[19] M.N. Mami, D. Graux, S. Scerri, H. Jabeen, S. Auer, J. Lehmann, Squerall: Virtual ontology-based access to heterogeneous and large data sources, in: Proc. ISWC, in: LNCS, Vol. 11779, Springer, 2019, pp. 229–245, http://dx.doi.org/10.1007/978-3-030-30796-7_15.

[20] S. Galhotra, U. Khurana, Semantic search over structured data, in: Proceedings of the 29th ACM International Conference on Information & Knowledge Management, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450368599, 2020, pp. 3381–3384, http://dx.doi.org/10.1145/3340531.3417426.

[21] P.N. Sawadogo, J. Darmont, On data lake architectures and metadata management, J. Intell. Inf. Syst. 56 (1) (2021) 97–120, http://dx.doi.org/10.1007/s10844-020-00608-7.

[22] R. Hai, C. Koutras, C. Quix, M. Jarke, Data lakes: A survey of functions and systems, IEEE Trans. Knowl. Data Eng. (2023) 1–20, http://dx.doi.org/10.1109/TKDE.2023.3270101.

[23] A. Adamou, M. d'Aquin, Linked data principles for data lakes, Data Lakes 2 (2020) 145–169.

[24] K. Węcel, Linked data for enrichment of data assets, in: Big, Open and Linked Data: Effects and Value for the Economy, Springer, 2022, pp. 35–71.

[25] J.C. Couto, D.D. Ruiz, An overview about data integration in data lakes, in: 2022 17th Iberian Conference on Information Systems and Technologies, CISTI, 2022, pp. 1–7, http://dx.doi.org/10.23919/CISTI54924.2022.9820576.

[26] G. Xiao, L. Ding, B. Cogrel, D. Calvanese, Virtual knowledge graphs: An overview of systems and use cases, Data Intell. 1 (3) (2019) 201–223.

[27] J. Liu, Y. Chabot, R. Troncy, V.-P. Huynh, T. Labbé, P. Monnin, From tabular data to knowledge graphs: A survey of semantic table interpretation tasks and methods, J. Web Semant. (2022) 100761.

[28] A. Chessa, G. Fenu, E. Motta, F. Osborne, D.R. Recupero, A.A. Salatino, L. Secchi, Enriching data lakes with knowledge graphs (short paper), in: Proc. Workshop on Knowledge Graph Generation from Text, in: CEUR WS, Vol. 3184, 2022, pp. 123–131, URL http://ceur-ws.org/Vol-3184/TEXT2KG_Short_1.pdf.

[29] É. Scholly, P.N. Sawadogo, P. Liu, J. Espinosa-Oviedo, C. Favre, S. Loudcher, J. Darmont, C. Noûs, Coining goldmedal: A new contribution to data lake generic metadata modeling, in: Proc. DOLAP, in: CEUR WS, Vol. 2840, 2021, pp. 31–40, URL http://ceur-ws.org/Vol-2840/paper5.pdf.

[30] D. Zhang, Y. Suhara, J. Li, M. Hulsebos, Ç. Demiralp, W. Tan, Sato: Contextual semantic type detection in tables, Proc. VLDB Endow. 13 (11) (2020) 1835–1848, URL http://www.vldb.org/pvldb/vol13/p1835-zhang.pdf.

[31] C. Quix, R. Hai, I. Vatov, Metadata extraction and management in data lakes with GEMMS, Complex Syst. Inform. Model. Q. 9 (2016) 67–83.

[32] R.C. Fernandez, Z. Abedjan, F. Koko, G. Yuan, S. Madden, M. Stonebraker, Aurum: A data discovery system, in: Proc. IEEE ICDE, 2018, pp. 1001–1012, http://dx.doi.org/10.1109/ICDE.2018.00094.

[33] Y. Suhara, J. Li, Y. Li, D. Zhang, Ç. Demiralp, C. Chen, W. Tan, Annotating columns with pre-trained language models, in: Proc. ACM SIGMOD, 2022, pp. 1493–1503.

[34] F. Nargesian, E. Zhu, R.J. Miller, K.Q. Pu, P.C. Arocena, Data lake management: challenges and opportunities, Proc. VLDB Endow. (ISSN: 2150-8097) 12 (12) (2019) 1986–1989, http://dx.doi.org/10.14778/3352063.3352116.

[35] S. Ji, S. Pan, E. Cambria, P. Marttinen, P.S. Yu, A survey on knowledge graphs: Representation, acquisition, and applications, IEEE Trans. Neural Netw. Learn. Syst. 33 (2) (2022) 494–514, http://dx.doi.org/10.1109/TNNLS.2021.3070843.

[36] M. Hulsebos, K.Z. Hu, M.A. Bakker, E. Zgraggen, A. Satyanarayan, T. Kraska, Ç. Demiralp, C.A. Hidalgo, Sherlock: A deep learning approach to semantic data type detection, in: Proc. ACM SIGKDD, 2019, pp. 1500–1508, http://dx.doi.org/10.1145/3292500.3330993.

[37] A. Burgdorf, A. Paulus, A. Pomp, T. Meisen, VC-SLAM - a handcrafted data corpus for the construction of semantic models, Data 7 (2) (2022) 17, http://dx.doi.org/10.3390/data7020017.

[38] C. Pinkel, C. Binnig, E. Jiménez-Ruiz, E. Kharlamov, A. Nikolov, A. Schwarte, C. Heupel, T. Kraska, IncMap: A journey towards ontology-based data integration, in: Proc. BTW, in: LNI, Vol. P-265, GI, 2017, pp. 145–164, URL https://dl.gi.de/20.500.12116/625.

[39] A. Paulus, A. Pomp, L. Poth, J. Lipp, T. Meisen, Gathering and combining semantic concepts from multiple knowledge bases, in: Proc. ICEIS, 2018, pp. 69–80, http://dx.doi.org/10.5220/0006800700690080.

[40] S.K. Ramnandan, A. Mittal, C.A. Knoblock, P.A. Szekely, Assigning semantic labels to data sources, in: Proc. ESWC, in: LNCS, Vol. 9088, Springer, 2015, pp. 403–417, http://dx.doi.org/10.1007/978-3-319-18818-8_25.

[41] M. Pham, S. Alse, C.A. Knoblock, P.A. Szekely, Semantic labeling: A domain-independent approach, in: Proc. ISWC, in: LNCS, Vol. 9981, 2016, pp. 446–462.

[42] N. Abdelmageed, S. Schindler, JenTab meets SemTab 2021's new challenges, in: Proc. SemTab, in: CEUR WS, Vol. 3103, 2021, pp. 42–53, URL http://ceur-ws.org/Vol-3103/paper4.pdf.

[43] X. Deng, H. Sun, A. Lees, Y. Wu, C. Yu, TURL: table understanding through representation learning, SIGMOD Rec. 51 (1) (2022) 33–40.

[44] M.N. Mami, S. Scerri, S. Auer, M. Vidal, Towards semantification of big data technology, in: Proc. DaWaK 2016, in: LNCS, Vol. 9829, Springer, 2016, pp. 376–390, http://dx.doi.org/10.1007/978-3-319-43946-4_25.

[45] A.Y. Halevy, F. Korn, N.F. Noy, C. Olston, N. Polyzotis, S. Roy, S.E. Whang, Goods: Organizing google's datasets, in: Proc. ACM SIGMOD, 2016, pp. 795–806, http://dx.doi.org/10.1145/2882903.2903730.

[46] A. Helal, M. Helali, K. Ammar, E. Mansour, A demonstration of kglac: A data discovery and enrichment platform for data science, Proc. VLDB Endow. 14 (12) (2021) 2675–2678, http://dx.doi.org/10.14778/3476311.3476317, URL http://www.vldb.org/pvldb/vol14/p2675-helal.pdf.

[47] A. Beheshti, B. Benatallah, R. Nouri, A. Tabebordbar, Corekg: a knowledge lake service, Proc. VLDB Endow. 11 (12) (2018) 1942–1945, http://dx.doi.org/10.14778/3229863.3236230, URL http://www.vldb.org/pvldb/vol11/p1942-beheshti.pdf.

[48] C. Stach, J. Bräcker, R. Eichler, C. Giebler, B. Mitschang, Demand-driven data provisioning in data lakes: BARENTS - a tailorable data preparation zone, in: Proc. IiWAS, ACM, 2021, pp. 187–198, http://dx.doi.org/10.1145/3487664.3487784.

[49] P.A. Szekely, C.A. Knoblock, J. Slepicka, A. Philpot, A. Singh, C. Yin, D. Kapoor, P. Natarajan, D. Marcu, K. Knight, D. Stallard, S.S. Karunamoorthy, R. Bojanapalli, S. Minton, B. Amanatullah, T. Hughes, M. Tamayo, D. Flynt, R. Artiss, S. Chang, T. Chen, G. Hiebel, L.S. Ferreira, Building and using a knowledge graph to combat human trafficking, in: Proc. ISWC, in: LNCS, Vol. 9367, Springer, 2015, pp. 205–221, http://dx.doi.org/10.1007/978-3-319-25010-6_12.

[50] A. Pomp, A. Paulus, A. Burgdorf, T. Meisen, A semantic data marketplace for easy data sharing within a smart city, in: Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Association for Computing Machinery, New York, NY, USA, ISBN: 9781450384469, 2021, pp. 4774–4778, http://dx.doi.org/10.1145/3459637.3481995.

[51] S. Nadal, K. Rabbani, O. Romero, S. Tadesse, ODIN: a dataspace management system, in: ISWC Satellite Tracks, in: CEUR WS, Vol. 2456, 2019, pp. 185–188, URL http://ceur-ws.org/Vol-2456/paper48.pdf.

[52] R. Eichler, C. Giebler, C. Gröger, H. Schwarz, B. Mitschang, Modeling metadata in data lakes - a generic model, Data Knowl. Eng. 136 (2021) 101931.

[53] C. Diamantini, P.L. Giudice, L. Musarella, D. Potena, E. Storti, D. Ursino, A new metadata model to uniformly handle heterogeneous data lake sources, in: Proc. ADBIS Short Papers & Workshops, in: CCIS, Vol. 909, Springer, 2018, pp. 165–177, http://dx.doi.org/10.1007/978-3-030-00063-9_17.

[54] C. Diamantini, D. Potena, E. Storti, A semantic data lake model for analytic query-driven discovery, in: Proc. IiWAS, ACM, 2021, pp. 183–186, http://dx.doi.org/10.1145/3487664.3487783.

[55] C. Diamantini, D. Potena, E. Storti, A knowledge-based approach to support analytic query answering in semantic data lakes, in: European Conference on Advances in Databases and Information Systems, Springer, 2022, pp. 179–192.

[56] A. Bagozi, D. Bianchini, V.D. Antonellis, M. Garda, M. Melchiori, Personalised exploration graphs on semantic data lakes, in: Proc. OTM Conf., in: LNCS, Vol. 11877, Springer, 2019, pp. 22–39, http://dx.doi.org/10.1007/978-3-030-33246-4_2.

[57] D. Bianchini, V.D. Antonellis, M. Garda, M. Melchiori, Exploiting smart city ontology and citizens' profiles for urban data exploration, in: Proc. OTM Conf., in: LNCS, Vol. 11229, Springer, 2018, pp. 372–389, http://dx.doi.org/10.1007/978-3-030-02610-3_21.

[58] D. Bianchini, V.D. Antonellis, M. Garda, M. Melchiori, Contextual preferences to personalise semantic data lake exploration, in: Proc. DEXA, in: LNCS, Vol. 12392, Springer, 2020, pp. 322–332, http://dx.doi.org/10.1007/978-3-030-59051-2_22.

[59] E.G. Kalayci, I. Grangel-González, F. Lösch, G. Xiao, A. ul Mehdi, E. Kharlamov, D. Calvanese, Semantic integration of bosch manufacturing data using virtual knowledge graphs, in: Proc. ISWC, in: LNCS, Vol. 12507, Springer, 2020, pp. 464–481.

[60] H. Dibowski, S. Schmid, Using knowledge graphs to manage a data lake, 2021.

[61] D. Calvanese, L. Ding, A. Mosca, G. Xiao, Realizing ontology-based reusable interfaces for data access via virtual knowledge graphs, in: Proc. Italian SIGCHI (CHItaly), ACM, 2021, pp. 35:1–35:5, http://dx.doi.org/10.1145/3464385.3464744.

[62] D. Brickley, M. Burgess, N.F. Noy, Google dataset search: Building a search engine for datasets in an open web ecosystem, in: L. Liu, R.W. White, A. Mantrach, F. Silvestri, J.J. McAuley, R. Baeza-Yates, L. Zia (Eds.), The World Wide Web Conference, WWW, San Francisco, CA, USA, ACM, 2019, pp. 1365–1375, http://dx.doi.org/10.1145/3308558.3313685.

[63] N.F. Noy, Y. Gao, A. Jain, A. Narayanan, A. Patterson, J. Taylor, Industry-scale knowledge graphs: lessons and challenges, Commun. ACM 62 (8) (2019) 36–43, http://dx.doi.org/10.1145/3331166.

[64] P. Liu, S. Loudcher, J. Darmont, E. Perrin, J. Girard, M. Rousset, Metadata model for an archeological data lake, in: Proc. Digital Humanities Conf., 2020.

[65] M. Zaharia, A. Ghodsi, R. Xin, M. Armbrust, Lakehouse: A new generation of open platforms that unify data warehousing and advanced analytics, in: Proc. CIDR, 2021, URL http://cidrdb.org/cidr2021/papers/cidr2021_paper17.pdf.

[66] G.A. Braun, L.A. Cecchi, P.R. Fillottrani, Towards *All-In-One* OBDA systems, in: Proc. Styrian Autumn of Ontology, in: CEUR WS, Vol. 2518, 2019, URL http://ceur-ws.org/Vol-2518/paper-DAOSI1.pdf.

[67] C. Pinkel, C. Binnig, E. Jiménez-Ruiz, E. Kharlamov, W. May, A. Nikolov, A.S. Bastinos, M.G. Skjæveland, A. Solimando, M. Taheriyan, C. Heupel, I. Horrocks, RODI: benchmarking relational-to-ontology mapping generation quality, Semantic Web 9 (1) (2018) 25–52, http://dx.doi.org/10.3233/SW-170268.

[68] S. Gottschalk, E. Demidova, Tab2KG: Semantic table interpretation with lightweight semantic profiles, Semantic Web (2022) 1–27.

[69] B. Vu, C.A. Knoblock, J. Pujara, Learning semantic models of data sources using probabilistic graphical models, in: Proc. WWW, 2019, pp. 1944–1953, http://dx.doi.org/10.1145/3308558.3313711.

[70] B. Vu, C.A. Knoblock, P.A. Szekely, M. Pham, J. Pujara, A graph-based approach for inferring semantic descriptions of wikipedia tables, in: Proc. ISWC, in: LNCS, Vol. 12922, Springer, 2021, pp. 304–320, http://dx.doi.org/10.1007/978-3-030-88361-4_18.

[71] R. Avogadro, M. Cremaschi, MantisTable V: a novel and efficient approach to semantic table interpretation, in: Proc. SemTab, in: CEUR WS, Vol. 3103, 2021, pp. 79–91, URL http://ceur-ws.org/Vol-3103/paper7.pdf.

[72] R. Shigapov, P. Zumstein, J. Kamlah, L. Oberländer, J. Mechnich, I. Schumm, Bbw: Matching CSV to wikidata via meta-lookup, in: Proc. SemTab 2020, in: CEUR WS, Vol. 2775, 2020, pp. 17–26, URL http://ceur-ws.org/Vol-2775/paper2.pdf.

[73] P. Nguyen, I. Yamada, N. Kertkeidkachorn, R. Ichise, H. Takeda, MTab4Wikidata at SemTab 2020: Tabular data annotation with wikidata, in: Proc. SemTab, in: CEUR WS, Vol. 2775, 2020, pp. 86–95, URL http://ceur-ws.org/Vol-2775/paper9.pdf.

[74] Z. Feng, W. Mayer, M. Stumptner, G. Grossmann, S. Kwashie, D. Ning, K. He, ASMaaS: Automatic semantic modeling as a service, in: Proc. IEEE SERVICES, 2021, pp. 33–40, http://dx.doi.org/10.1109/SERVICES51467.2021.00033.

[75] G. Futia, A. Vetrò, J.C.D. Martin, SeMi: A semantic modeling machine to build knowledge graphs with graph neural networks, SoftwareX 12 (2020) 100516, http://dx.doi.org/10.1016/j.softx.2020.100516.

[76] M. Taheriyan, C.A. Knoblock, P.A. Szekely, J.L. Ambite, A graph-based approach to learn semantic descriptions of data sources, in: Proc. ISWC, in: LNCS, Vol. 8218, Springer, 2013, pp. 607–623, http://dx.doi.org/10.1007/978-3-642-41335-3_38.

[77] M. Taheriyan, C.A. Knoblock, P.A. Szekely, J.L. Ambite, Leveraging linked data to discover semantic relations within data sources, in: Proc. ISWC, in: LNCS, Vol. 9981, 2016, pp. 549–565, http://dx.doi.org/10.1007/978-3-319-46523-4_33.

[78] M. Taheriyan, C.A. Knoblock, P.A. Szekely, J.L. Ambite, Learning the semantics of structured data sources, J. Web Semant. 37–38 (2016) 152–169, http://dx.doi.org/10.1016/j.websem.2015.12.003.

[79] C.A. Knoblock, P.A. Szekely, J.L. Ambite, A. Goel, S. Gupta, K. Lerman, M. Muslea, M. Taheriyan, P. Mallick, Semi-automatically mapping structured sources into the semantic web, in: Proc. ESWC, in: LNCS, Vol. 7295, Springer, 2012, pp. 375–390, http://dx.doi.org/10.1007/978-3-642-30284-8_32.

[80] A. Burgdorf, A. Paulus, A. Pomp, T. Meisen, DocSemMap: Leveraging textual data documentations for mapping structured data sets into knowledge graphs, in: Proc. IEEE ICSC, 2022, pp. 209–216, http://dx.doi.org/10.1109/ICSC52841.2022.00042.

[81] M. Ramirez, A. Bogatu, N.W. Paton, A. Freitas, Natural language inference over tables: Enabling explainable data exploration on data lakes, in: Proc. ESWC, in: LNCS, Vol. 12731, Springer, 2021, pp. 304–320, http://dx.doi.org/10.1007/978-3-030-77385-4_18.

[82] R.C. Fernandez, E. Mansour, A.A. Qahtan, A.K. Elmagarmid, I.F. Ilyas, S. Madden, M. Ouzzani, M. Stonebraker, N. Tang, Seeping semantics: Linking datasets using word embeddings for data discovery, in: Proc. IEEE ICDE, 2018, pp. 989–1000, http://dx.doi.org/10.1109/ICDE.2018.00093.

[83] D. Haller, R. Lenz, Pharos: Query-driven schema inference for the semantic web, in: ECML/PKDD Workshops, Part II, in: CCIS, Vol. 1168, Springer, 2019, pp. 112–124, http://dx.doi.org/10.1007/978-3-030-43887-6_10.

[84] D. Haller, R. Lenz, Discovery of ontologies from implicit user knowledge, in: Proc. Conf. on "Lernen, Wissen, Daten, Analysen", in: CEUR WS, Vol. 2738, 2020, pp. 241–245, URL http://ceur-ws.org/Vol-2738/LWDA2020_paper_4.pdf.

[85] A. Pomp, A. Paulus, S. Jeschke, T. Meisen, ESKAPE: information platform for enabling semantic data processing, in: Proc. ICEIS, SciTePress, 2017, pp. 644–655, http://dx.doi.org/10.5220/0006324906440655.

[86] S. Gupta, P.A. Szekely, C.A. Knoblock, A. Goel, M. Taheriyan, M. Muslea, Karma: A system for mapping structured sources into the semantic web, in: ESWC Satellite Events - Revised Selected Papers, in: LNCS, Vol. 7540, Springer, 2012, pp. 430–434, http://dx.doi.org/10.1007/978-3-662-46641-4_40.

[87] A. Pomp, J. Lipp, T. Meisen, Enabling the continuous evolution of ontologies for ontology-based data management, Int. J. Robot. Comput. (2019).

[88] Y. Chabot, T. Labbé, J. Liu, R. Troncy, DAGOBAH: an end-to-end context-free tabular data semantic annotation system, in: Proc. SemTab, in: CEUR WS, Vol. 2553, 2019, pp. 41–48, URL http://ceur-ws.org/Vol-2553/paper6.pdf.

[89] P. Nguyen, N. Kertkeidkachorn, R. Ichise, H. Takeda, Mtab: Matching tabular data to knowledge graph with probability models, in: Proc. Workshop on Ontology Matching, in: CEUR WS, Vol. 2536, 2019, pp. 191–192, URL http://ceur-ws.org/Vol-2536/om2019_poster2.pdf.

[90] N. Abdelmageed, S. Schindler, JenTab: A toolkit for semantic table annotations, in: Proc. Workshop on Knowledge Graph Construction, in: CEUR WS, Vol. 2873, 2021, URL http://ceur-ws.org/Vol-2873/paper5.pdf.

[91] M.N. Mami, I. Grangel-González, D. Graux, E. Elezi, F. Lösch, Semantic data integration for the SMT manufacturing process using SANSA stack, in: ESWC Satellite Events, Revised Selected Papers, in: LNCS, Vol. 12124, Springer, 2020, pp. 307–311, http://dx.doi.org/10.1007/978-3-030-62327-2_47.

[92] L. Gagliardelli, L. Zecchini, D. Beneventano, G. Simonini, S. Bergamaschi, M. Orsini, M. Luca, M. Emma, L. Andrea, G. Nicola, et al., ECDP: A big data platform for the smart monitoring of local energy communities, in: CEUR Workshop Proceedings, Vol. 3135, 2022.

[93] A. Poggi, D. Lembo, D. Calvanese, G.D. Giacomo, M. Lenzerini, R. Rosati, Linking data to ontologies, J. Data Semant. 10 (2008) 133–173.

[94] D. Calvanese, G.D. Giacomo, D. Lembo, M. Lenzerini, R. Rosati, Ontology-based data access and integration, in: L. Liu, M.T. Özsu (Eds.), Encyclopedia of Database Systems, second ed., Springer, 2018, http://dx.doi.org/10.1007/978-1-4614-8265-9_80667.

[95] N. Fathy, W. Gad, N. Badr, A unified access to heterogeneous big data through ontology-based semantic integration, in: Proc. ICICIS, IEEE, 2019, pp. 387–392.

[96] P. Jovanovic, S. Nadal, O. Romero, A. Abelló, B. Bilalli, Quarry: A user-centered big data integration platform, Inf. Syst. Front. 23 (1) (2021) 9–33, http://dx.doi.org/10.1007/s10796-020-10001-y.

[97] E. Iglesias, S. Jozashoori, D. Chaves-Fraga, D. Collarana, M. Vidal, SDM-RDFizer: An RML interpreter for the efficient creation of RDF knowledge graphs, in: Proc. ACM CIKM, 2020, pp. 3039–3046.

[98] E. Daga, L. Asprino, P. Mulholland, A. Gangemi, Facade-X: an opinionated approach to SPARQL anything, 2021, arXiv arXiv:2106.02361, URL https://api.semanticscholar.org/CorpusID:235353029.

[99] Z. Gu, F. Corcoglioniti, D. Lanti, A. Mosca, G. Xiao, J. Xiong, D. Calvanese, A systematic overview of data federation systems, Semantic Web 15 (1) (2024) 107–165, https://doi.org/10.3233/SW-223201.

[100] M. Lenzerini, Data integration: A theoretical perspective, in: Proc. ACM PODS, 2002, pp. 233–246, http://dx.doi.org/10.1145/543613.543644.

[101] M.N. Mami, D. Graux, H. Thakkar, S. Scerri, S. Auer, J. Lehmann, The query translation landscape: a survey, 2019, CoRR arXiv:1910.03118, URL http://arxiv.org/abs/1910.03118.

[102] D. Calvanese, G.D. Giacomo, D. Lembo, M. Lenzerini, A. Poggi, M. Rodriguez-Muro, R. Rosati, M. Ruzzi, D.F. Savo, The MASTRO system for ontology-based data access, Semantic Web 2 (1) (2011) 43–53, http://dx.doi.org/10.3233/SW-2011-0029.

[103] J.F. Sequeda, D.P. Miranker, Ultrawrap: SPARQL execution on relational data, J. Web Semant. 22 (2013) 19–39, http://dx.doi.org/10.1016/j.websem.2013.08.002.

[104] F. Priyatna, Ó. Corcho, J.F. Sequeda, Formalisation and experiences of R2RML-based SPARQL to SQL query translation using morph, in: Proc. WWW, ACM, 2014, pp. 479–490, http://dx.doi.org/10.1145/2566486.2567981.

[105] E. Kharlamov, E. Jiménez-Ruiz, D. Zheleznyakov, D. Bilidas, M. Giese, P. Haase, I. Horrocks, H. Kllapi, M. Koubarakis, Ö.L. Özçep, M. Rodriguez-Muro, R. Rosati, M. Schmidt, R. Schlatte, A. Soylu, A. Waaler, Optique: Towards OBDA systems for industry, in: Proc. ESWC Satellite Events, Revised Selected Papers, in: LNCS, Vol. 7955, Springer, 2013, pp. 125–140, http://dx.doi.org/10.1007/978-3-642-41242-4_11.

[106] Y. Chronis, Y. Foufoulas, V. Nikolopoulos, A. Papadopoulos, L. Stamatogiannakis, C. Svingos, Y.E. Ioannidis, A relational approach to complex dataflows, in: Proc. EDBT/ICDT Workshops, in: CEUR WS, 1558, 2016, URL http://ceur-ws.org/Vol-1558/paper45.pdf.

[107] T. Bagosi, D. Calvanese, J. Hardi, S. Komla-Ebri, D. Lanti, M. Rezk, M. Rodriguez-Muro, M. Slusnys, G. Xiao, The ontop framework for ontology based data access, in: Proc. CSWS, Revised Selected Papers, in: CCIS, Vol. 480, Springer, 2014, pp. 67–77, http://dx.doi.org/10.1007/978-3-662-45495-4_6.

[108] G. Xiao, D. Lanti, R. Kontchakov, S. Komla-Ebri, E.G. Kalayci, L. Ding, J. Corman, B. Cogrel, D. Calvanese, E. Botoeva, The virtual knowledge graph system ontop (extended abstract), in: Proc. Description Logics Workshop, in: CEUR WS, Vol. 2663, 2020, URL http://ceur-ws.org/Vol-2663/abstract-23.pdf.

[109] E. Kharlamov, D. Hovland, M.G.S. veland, D. Bilidas, E. Jiménez-Ruiz, G. Xiao, A. Soylu, D. Lanti, M. Rezk, D. Zheleznyakov, M. Giese, H. Lie, Y.E. Ioannidis, Y. Kotidis, M. Koubarakis, A. Waaler, Ontology based data access in statoil, J. Web Semant. 44 (2017) 3–36, http://dx.doi.org/10.1016/j.websem.2017.05.005.

[110] M. Mansfield, V.A.M. Tamma, P. Goddard, F. Coenen, Capturing expert knowledge for building enterprise SME knowledge graphs, in: Proc. K-CAP, ACM, 2021, pp. 129–136, http://dx.doi.org/10.1145/3460210.3493569.

[111] D. Bilidas, M. Koubarakis, Efficient duplicate elimination in SPARQL to SQL translation, in: Proc. Description Logics Workshop, in: CEUR WS, Vol. 2211, 2018, URL http://ceur-ws.org/Vol-2211/paper-08.pdf.

[112] P.D. Rohde, M. Vidal, Optimizing federated queries based on the physical design of a data lake, in: Proc. EDBT/ICDT Workshops, in: CEUR WS, Vol. 2578, 2020, URL http://ceur-ws.org/Vol-2578/SEAData6.pdf.

[113] D. Chaves-Fraga, E. Ruckhaus, F. Priyatna, M. Vidal, Ó. Corcho, Enhancing virtual ontology based access over tabular data with Morph-CSV, Semantic Web 12 (6) (2021) 869–902, http://dx.doi.org/10.3233/SW-210432.

[114] K. Bereta, G. Papadakis, M. Koubarakis, Ontop4theWeb: Sparqling the web on-the-fly, in: Proc. IEEE ICSC, 2021, pp. 268–271, http://dx.doi.org/10.1109/ICSC50631.2021.00053.

[115] F. Schwade, P. Schubert, A semantic data lake for harmonizing data from cross-platform digital workspaces using ontology-based data access, in: Proc. AMCIS, 2020, URL https://aisel.aisnet.org/amcis2020/ai_semantic_for_intelligent_info_systems/ai_semantic_for_intelligent_info_systems/2.

[116] M. Belcao, E. Falzone, E. Bionda, E.D. Valle, Chimera: A bridge between big data analytics and semantic technologies, in: Proc. ISWC, in: LNCS, Vol. 12922, Springer, 2021, pp. 463–479, http://dx.doi.org/10.1007/978-3-030-88361-4_27.

[117] K.M. Endris, P.D. Rohde, M. Vidal, S. Auer, Ontario: Federated query processing against a semantic data lake, in: Proc. DEXA, in: LNCS, Vol. 11706, Springer, 2019, pp. 379–395, http://dx.doi.org/10.1007/978-3-030-27615-7_29.

[118] M. Buron, F. Goasdoué, I. Manolescu, M. Mugnier, Obi-wan: Ontology-based RDF integration of heterogeneous data, Proc. VLDB Endow. 13 (12) (2020) 2933–2936, http://dx.doi.org/10.14778/3415478.3415512, URL http://www.vldb.org/pvldb/vol13/p2933-buron.pdf.

[119] R. Bonaque, T.D. Cao, B. Cautis, F. Goasdoué, J. Letelier, I. Manolescu, O. Mendoza, S. Ribeiro, X. Tannier, M. Thomazo, Mixed-instance querying: a lightweight integration architecture for data journalism, Proc. VLDB Endow. 9 (13) (2016) 1513–1516, http://dx.doi.org/10.14778/3007263.3007297, URL http://www.vldb.org/pvldb/vol9/p1513-bonaque.pdf.

[120] J. Baget, M. Leclère, M. Mugnier, S. Rocher, C. Sipieter, Graal: A toolkit for query answering with existential rules, in: Proc. RuleML, in: LNCS, Vol. 9202, Springer, 2015, pp. 328–344, http://dx.doi.org/10.1007/978-3-319-21542-6_21.

[121] S. Nadal, A. Abelló, O. Romero, S. Vansummeren, P. Vassiliadis, Graph-driven federated data management, IEEE Trans. Knowl. Data Eng. 35 (1) (2023) 509–520, http://dx.doi.org/10.1109/TKDE.2021.3077044.

[122] Y. Khan, A. Zimmermann, A. Jha, V. Gadepally, M. d'Aquin, R. Sahay, One size does not fit all: Querying web polystores, IEEE Access 7 (2019) 9598–9617, http://dx.doi.org/10.1109/ACCESS.2018.2888601.

[123] O. Curé, R. Hecht, C.L. Duc, M. Lamolle, Data integration over NoSQL stores using access path based mappings, in: Proc. DEXA, in: LNCS, Vol. 6860, Springer, 2011, pp. 481–495, http://dx.doi.org/10.1007/978-3-642-23088-2_36.

[124] O. Curé, F. Kerdjoudj, D. Faye, C.L. Duc, M. Lamolle, On the potential integration of an ontology-based data access approach in NoSQL stores, Int. J. Distributed Syst. Technol. 4 (3) (2013) 17–30, http://dx.doi.org/10.4018/jdst.2013070102.

[125] A. Dimou, M.V. Sande, P. Colpaert, R. Verborgh, E. Mannens, R.V. de Walle, RML: a generic language for integrated RDF mappings of heterogeneous data, in: Proc. LDOW Workshop, in: CEUR WS, Vol. 1184, 2014, URL http://ceur-ws.org/Vol-1184/ldow2014_paper_01.pdf.

[126] B.D. Meester, W. Maroy, A. Dimou, R. Verborgh, E. Mannens, Declarative data transformations for linked data generation: The case of DBpedia, in: Proc. ESWC, in: LNCS, Vol. 10250, 2017, pp. 33–48, http://dx.doi.org/10.1007/978-3-319-58451-5_3.

[127] A.C. Junior, C. Debruyne, R. Brennan, D. O'Sullivan, FunUL: a method to incorporate functions into uplift mapping languages, in: Proc. IiWAS, ACM, 2016, pp. 267–275, http://dx.doi.org/10.1145/3011141.3011152.

[128] P. Heyvaert, B.D. Meester, A. Dimou, R. Verborgh, Declarative rules for linked data generation at your fingertips!, in: ESWC Satellite Events, Revised Selected Papers, in: LNCS, Vol. 11155, Springer, 2018, pp. 213–217, http://dx.doi.org/10.1007/978-3-319-98192-5_40.

[129] P. Heyvaert, A. Dimou, A. Herregodts, R. Verborgh, D. Schuurman, E. Mannens, R.V. de Walle, Rmleditor: A graph-based mapping editor for linked data mappings, in: Proc. ESWC, in: LNCS, Vol. 9678, Springer, 2016, pp. 709–723, http://dx.doi.org/10.1007/978-3-319-34129-3_43.

[130] P.R. Aryan, F.J. Ekaputra, E. Kiesling, A.M. Tjoa, K. Kurniawan, RMLx: Mapping interface for integrating open data with linked data exploration environment, in: Proc. ICICoS, IEEE, 2017, pp. 113–118.

[131] Á. Sicilia, G. Nemirovski, A. Nolle, Map-on: A web-based editor for visual ontology mapping, Semantic Web 8 (6) (2017) 969–980, http://dx.doi.org/10.3233/SW-160246.

[132] P. Heyvaert, A. Dimou, B.D. Meester, T. Seymoens, A. Herregodts, R. Verborgh, D. Schuurman, E. Mannens, Specification and implementation of mapping rule visualization and editing: MapVOWL and the RMLEditor, J. Web Semant. 49 (2018) 31–50, http://dx.doi.org/10.1016/j.websem.2017.12.003.

[133] H. García-González, I. Boneva, S. Staworko, J.E.L. Gayo, J.M.C. Lovelle, ShExML: improving the usability of heterogeneous data mapping languages for first-time users, PeerJ Comput. Sci. 6 (2020) e318, http://dx.doi.org/10.7717/peerj-cs.318.

[134] M. Lefrançois, A. Zimmermann, N. Bakerally, Flexible RDF generation from RDF and heterogeneous data sources with SPARQL-generate, in: Proc. EKAW Satellite Events, Revised Selected Papers, in: LNCS, Vol. 10180, Springer, 2016, pp. 131–135, http://dx.doi.org/10.1007/978-3-319-58694-6_16.

[135] U. Simsek, E. Kärle, D. Fensel, Rocketrml - a nodejs implementation of a use case specific RML mapper, in: Proc. ESWC Workshops, in: CEUR WS, 2489, 2019, pp. 46–53, URL http://ceur-ws.org/Vol-2489/paper5.pdf.

[136] F. Michel, C. Faron-Zucker, J. Montagnat, Bridging the semantic web and NoSQL worlds: Generic SPARQL query translation and application to mongodb, Trans. Large Scale Data Knowl. Centered Syst. 40 (2019) 125–165.

[137] T. Delva, D.V. Assche, P. Heyvaert, B.D. Meester, A. Dimou, Integrating nested data into knowledge graphs with RML fields, in: Proc. Knowledge Graph Construction Workshop, in: CEUR WS, Vol. 2873, 2021, URL http://ceur-ws.org/Vol-2873/paper9.pdf.

[138] T.H.D. Araujo, B.T. Agena, K.R. Braghetto, R. Wassermann, OntoMongo – ontology-based data access for NoSQL, in: Proc. IX Seminar on Ontology Research, in: CEUR WS, Vol. 1908, 2017, pp. 55–66, URL http://ceur-ws.org/Vol-1908/paper5.pdf.

[139] E. Botoeva, D. Calvanese, B. Cogrel, J. Corman, G. Xiao, Ontology-based data access - beyond relational sources, Intell. Artif. 13 (1) (2019) 21–36, http://dx.doi.org/10.3233/IA-190023.

[140] H. El Massari, S. Mhammedi, N. Gherabi, M. Nasri, Virtual OBDA mechanism ontop for answering SPARQL queries over couchbase, in: Proc. ICATH, Springer, 2022, pp. 193–205.

[141] N. Fathy, W.K. Gad, N.L. Badr, M. Hashem, Querying heterogeneous property graph data sources based on a unified conceptual view, in: Proc. ICSIE, ACM, 2020, pp. 113–118, http://dx.doi.org/10.1145/3436829.3436855.

[142] M.-L. Mugnier, M.-C. Rousset, F. Ulliana, Ontology-mediated queries for NOSQL databases, in: Proc. AAAI, 2016.

[143] J. Lehmann, G. Sejdiu, L. Bühmann, P. Westphal, C. Stadler, I. Ermilov, S. Bin, N. Chakraborty, M. Saleem, A.N. Ngomo, H. Jabeen, Distributed semantic analytics using the SANSA stack, in: Proc. ISWC, in: LNCS, Vol. 10588, Springer, 2017, pp. 147–155, http://dx.doi.org/10.1007/978-3-319-68204-4_15.

[144] M.N. Mami, D. Graux, S. Scerri, H. Jabeen, S. Auer, Querying data lakes using spark and presto, in: L. Liu, R.W. White, A. Mantrach, F. Silvestri, J.J. McAuley, R. Baeza-Yates, L. Zia (Eds.), The World Wide Web Conference, WWW, San Francisco, CA, USA, ACM, 2019, pp. 3574–3578, http://dx.doi.org/10.1145/3308558.3314132.

[145] A. Chortaras, G. Stamou, D2RML: integrating heterogeneous data and web services into custom RDF graphs, in: Proc. LDOW Workshop, in: CEUR WS, Vol. 2073, 2018, URL http://ceur-ws.org/Vol-2073/article-07.pdf.

[146] Ó. Corcho, F. Priyatna, D. Chaves-Fraga, Towards a new generation of ontology based data access, Semantic Web 11 (1) (2020) 153–160, http://dx.doi.org/10.3233/SW-190384.

[147] L.F. de Medeiros, F. Priyatna, Ó. Corcho, MIRROR: automatic R2RML mapping generation from relational databases, in: Proc. ICWE, in: LNCS, Vol. 9114, Springer, 2015, pp. 326–343, http://dx.doi.org/10.1007/978-3-319-19890-3_21.

[148] M.A.G. Hazber, R. Li, G. Xu, K.M. Alalayah, An approach for automatically generating R2RML-based direct mapping from relational databases, in: Proc. ICYCSEE, in: CCIS, Vol. 623, Springer, 2016, pp. 151–169, http://dx.doi.org/10.1007/978-981-10-2053-7_15.

[149] P. Heyvaert, A. Dimou, R. Verborgh, E. Mannens, Ontology-based data access mapping generation using data, schema, query, and mapping knowledge, in: Proc. ESWC, Vol. 10250, 2017, pp. 205–215, http://dx.doi.org/10.1007/978-3-319-58451-5_15.

[150] Á. Sicilia, G. Nemirovski, AutoMap4OBDA: Automated generation of R2RML mappings for OBDA, in: Proc. EKAW, in: LNCS, Vol. 10024, 2016, pp. 577–592, http://dx.doi.org/10.1007/978-3-319-49004-5_37.

[151] S. Mathur, D. O'Sullivan, R. Brennan, Milan: Automatic generation of R2RML mappings, in: Proc. AIAI Irish Conf. on AI & Cognitive Science, in: CEUR WS, Vol. 2259, 2018, pp. 90–101, URL http://ceur-ws.org/Vol-2259/aics_10.pdf.

[152] A. Iglesias-Molina, L. Pozo-Gilo, D. Doña, E. Ruckhaus, D. Chaves-Fraga, Ó. Corcho, Mapeathor: Simplifying the specification of declarative rules for knowledge graph construction, in: Proc. ISWC Demos Track, Vol. 2721, 2020, pp. 25–30.

[153] B. Zhou, Y. Svetashova, A. Gusmao, A. Soylu, G. Cheng, R. Mikut, A. Waaler, E. Kharlamov, SemML: Facilitating development of ML models for condition monitoring with semantics, J. Web Semant. 71 (2021) 100664, http://dx.doi.org/10.1016/j.websem.2021.100664.

[154] A. Ustundag, E. Cevikcan, Industry 4.0: Managing the Digital Transformation, Springer, 2018.

[155] J. Strötgen, T. Tran, A. Friedrich, D. Milchevski, F. Tomazic, A. Marusczyk, H. Adel, D. Stepanova, F. Hildebrand, E. Kharlamov, Towards the bosch materials science knowledge base, in: Proc. ISWC Satellite Tracks, in: CEUR WS, Vol. 2456, 2019, pp. 323–324, URL http://ceur-ws.org/Vol-2456/paper89.pdf.

[156] B. Zhou, D. Zhou, J. Chen, Y. Svetashova, G. Cheng, E. Kharlamov, Scaling usability of ML analytics with knowledge graphs: Exemplified with a bosch welding case, in: Proc. IJCKG, ACM, 2021, pp. 54–63, http://dx.doi.org/10.1145/3502223.3502230.

[157] B. Zhou, Y. Svetashova, T. Pychynski, I. Baimuratov, A. Soylu, E. Kharlamov, SemFE: Facilitating ML pipeline development with semantics, in: Proc. ACM CIKM, 2020, pp. 3489–3492, http://dx.doi.org/10.1145/3340531.3417436.

[158] A. Zanella, N. Bui, A. Castellani, L. Vangelista, M. Zorzi, Internet of things for smart cities, IEEE Int. Things J. 1 (1) (2014) 22–32.

[159] D. Bianchini, V. De Antonellis, M. Garda, M. Melchiori, Smart city data modelling using semantic web technologies, in: IEEE International Smart Cities Conference (ISC2), 2021, pp. 1–7, http://dx.doi.org/10.1109/ISC253183.2021.9562913.

[160] S. Nahhas, Potentials of semantic internet of things in smart cities: an overview and roadmap, in: 1st International Conference on Advanced Innovations in Smart Cities, ICAISC, 2023, pp. 1–6, http://dx.doi.org/10.1109/ICAISC56366.2023.10085121.

[161] K. Korini, C. Bizer, Column type annotation using ChatGPT, 2023, arXiv preprint arXiv:2306.00745.