Large-scale Semantic Integration of Linked Data: A Survey

MICHALIS MOUNTANTONAKIS and YANNIS TZITZIKAS, Institute of Computer Science, FORTH-ICS, Greece & Computer Science Department, University of Crete, Greece

A large number of published datasets (or sources) that follow Linked Data principles is currently available and this number grows rapidly. However, the major target of Linked Data, i.e., linking and integration, is not easy to achieve. In general, information integration is difficult, because (a) datasets are produced, kept, or managed by different organizations using different models, schemas, or formats, (b) the same real-world entities or relationships are referred with different URIs or names and in different natural languages, (c) datasets usually contain complementary information, (d) datasets can contain data that are erroneous, out-of-date, or conflicting, (e) datasets even about the same domain may follow different conceptualizations of the domain, (f) everything can change (e.g., schemas, data) as time passes. This article surveys the work that has been done in the area of Linked Data integration, it identifies the main actors and use cases, it analyzes and factorizes the integration process according to various dimensions, and it discusses the methods that are used in each step. Emphasis is given on methods that can be used for integrating several datasets. Based on this analysis, the article concludes with directions that are worth further research.

CCS Concepts: • **Information systems → Information integration**; **Resource Description Framework (RDF)**;

Additional Key Words and Phrases: Data integration, data discovery, RDF, semantic web, big data

ACM Reference format:
Michalis Mountantonakis and Yannis Tzitzikas. 2019. Large-scale Semantic Integration of Linked Data: A Survey. *ACM Comput. Surv.* 52, 5, Article 103 (September 2019), 40 pages.
https://doi.org/10.1145/3345551

1 INTRODUCTION

The major target of Linked Open Data (LOD) is linking and integration for easing data discovery process, for performing data analysis, and for offering integrated query answering. However, we are still far from achieving this target. The integration process still requires a number of steps, some of which are difficult or costly. As it is stated in Reference [40], *"Integration requires spending resources on mapping heterogeneous data items, resolving conflicts, cleaning the data, and so on. Such costs can also be huge. Actually, the cost of integrating some sources may not be worthwhile if the gain is limited, especially in the presence of redundant data and low quality data."* Moreover, it has

The research work was supported by the Hellenic Foundation for Research and Innovation (HFRI) and the General Secretariat for Research and Technology (GSRT), under the HFRI PhD Fellowship Grant No. 166.
Authors' addresses: M. Mountantonakis and Y. Tzitzikas, Institute of Computer Science, FORTH-ICS, Greece & Computer Science Department, University of Crete, Greece, N. Plastira 100, Vassilika Vouton, GR-700 13 Heraklion, Crete, Greece; emails: {mountant, tzitzik}@ics.forth.gr.

0360-0300/2019/09-ART103 $15.00
https://doi.org/10.1145/3345551

been written[1] that *"Data scientists spend from 50 percent to 80 percent of their time in collecting and preparing unruly digital data, before it can be explored for useful nuggets."*

The integration is an inherently difficult problem for various reasons. Initially, datasets are produced and managed by different organizations according to conceptualizations, models, schemas, and formats based on their needs and choice, and these data are stored in different locations and systems. Another difficulty is that the same real-world entities or relationships are referred to with different names or URIs and in different natural languages; the synonyms and homonyms of natural languages further perplex identification and matching. Moreover, quite often there is not enough data for automated matching, because datasets contain complementary information about the same entities. Data integration also has to tackle the fact that datasets usually contain erroneous, out-of-date, or conflicting data. Finally, integration is not a one-shot task in the sense that everything can change as time passes (e.g., schemas, ontologies, naming conventions, etc.), necessitating additional actions for curating the integrated access. Compared to other technologies (e.g., relational databases), Linked Data (and RDF) use URIs instead of simple names, which is a solution for the problem of homonyms, while the problem of synonyms can be tackled by defining equivalence relationships among different concepts or entities (e.g., between two different URIs that describe the same entity). Finally, data that have been described by using different schemas can be transformed and integrated in a more flexible way, since RDF is a graph-based model.

Data integration has been studied in the context of various data models, namely, in the relational model, e.g., see Reference [65] for a survey of information integration in the relational model, and in the semi-structured model, e.g., see Reference [11] for a survey of approaches for integration for XML databases, while Reference [41] surveys works for big data integration for both models. In this article, we survey the work that has been done in the area of Linked Data integration. There are surveys for various individual tasks of the general integration process, i.e., surveys for *distributed RDF processing* [74], for *ontology matching* [144], for *instance matching* [108], for *integration for OLAP* [1], for *query federation* [130], for *visualization and exploration* [10, 29, 153], and for *quality* [119, 168]. In the current survey, we aim at covering the topic holistically, i.e., from various perspectives, for better understanding the overall problem and process, and for making more obvious the dependence between the individual tasks. Moreover, since the LOD cloud already contains a large number of datasets (over 9,000 datasets according to Reference [47]), we give emphasis on methods that can be applied to very large number of datasets. This distinction is important in the sense that a semi-automatic process that can be followed for integrating a few (say five) datasets, is not affordable, due to the required human effort, for integrating thousands of datasets. Michael Stonebraker (a pioneer researcher in data management) has mentioned (http://ilp.mit.edu/images/conferences/2013/ict/presentation/stonebraker.pdf) that data integration at scale is a very big deal and probably the biggest problem that many enterprises face, since the traditional approaches cannot scale easily to more than 25 sources. For this reason, in this survey we emphasize on tasks that could aid the integration of large number of datasets, and discuss the tools that are available for several RDF datasets. Overall this survey provides a concise overview of the issues, methods, tools and systems for semantic integration of data.

In various places of the article, we shall use a running example from the marine domain. The rest of this article is organized as follows: Section 2 provides the background and the context, discusses the Linked Data ecosystem and refers to past surveys. Section 3 introduces the difficulties of data integration, while Section 4 describes the landscape of data integration. Section 5 surveys the integration methods, while Section 6 discusses the different integration processes. Section 7

[1] http://www.nytimes.com/2014/08/18/technology/for-big-data-scientists-hurdle-to-insights-is-janitor-work.html?_r=0.

focuses on evaluation related to information integration, whereas Section 8 lists tools and services for several RDF datasets and what steps of the processes they cover and how. Section 9 identifies some research directions, and Section 10 concludes the article.

2 BACKGROUND AND CONTEXT

In Section 2.1, we introduce the main principles of Linked Data, in Section 2.2, we discuss the Linked Data Ecosystem, and in Section 2.3, we analyze related surveys.

2.1 Linked Data

Definition and Roots. "Linked Data refers to a method of publishing structured data, so that it can be interlinked and become more useful through semantic queries, founded on HTTP, RDF and URIs" [12]. In the 1990s, Tim Berners-Lee, the inventor of the World Wide Web, discussed the vision for a Semantic Web [8], i.e., *"The first step is putting data on the Web in a form that machines can naturally understand, or converting it to that form. This creates what I call a Semantic Web—a web of data that can be processed directly or indirectly by machines."* In 2001, Tim Berners-Lee and his colleagues described the main ideas of Semantic Web [9], e.g., representing data in RDF format, using ontologies that enable the creation of inference rules, and others, while the May 2006 paper [143], stressed the emerging need for semantic data integration and described most of the Linked Data principles, which are discussed below.

Linked Data Principles. The major principles of Linked Data, which are required for reaching the goal for a "Web of Data" (or Semantic Web) [12], were officially proposed in July 2006 by Tim Berners-Lee[2]: "(1) use URIs as names for things, (2) use HTTP URIs so that people can look up those names, (3) when someone looks up a URI, provide useful information, using the standards (RDF, SPARQL), and (4) include links to other URIs, so that they can discover more things." The fourth principle, which refers to data interlinking, is of primary importance for data integration, since it suggests to the publishers to create relationships with URIs occurring in different datasets. The datasets can be linked through common URIs, which can refer to either schema elements (they are defined through RDF Schema and OWL [5]) or data elements. SPARQL is a standard query language (http://www.w3.org/TR/sparql11-query/) for retrieving and managing RDF data, whereas queries can be expressed across different datasets. Moreover, the urge and tendency towards linking and integration can be observed from proposals for rating *open data*, i.e., by using the 5-star Open Data (http://5stardata.info/en/), as well as proposals for rating *vocabularies* [77].

Formally, the Resource Description Framework (`RDF`) is a "graph-based data model" [5]. In RDF, `Triples` are used for relating resources, i.e., Uniform Resource Identifiers (`URIs`) or anonymous ones (`blank nodes`), with other resources (i.e., `URIs` or `blank nodes`) or `Literals` (i.e., constants). We define the set of all URIs as U, the set of all `blank nodes` as B, whereas let L be the set of all Literals. Each triple is a statement of the following form: subject-predicate-object. A subject describes an entity, a predicate corresponds to a property of that entity, and an object corresponds to the value of the aforementioned property for the entity occuring as subject. For instance, in Figure 1, we can see an example of Linked Data with 9 triples. One triple is the following: ⟨Yellowfin_Tuna, livesInOcean, Pacific⟩ where the subject is Yellowfin_Tuna, livesInOcean corresponds to the predicate and Pacific is the object of that triple. Moreover, we define as S the set of all `subjects`, as P the set of all `properties`, whereas the set of all `objects` is denoted as O. A *triple* can be defined formally as any element of $T = S \times P \times O$, where $S = U \cup B$, $P = U$ and $O = U \cup B \cup L$. Finally, an *RDF graph* (or dataset) can be constructed by any finite subset of T.

[2]https://www.w3.org/DesignIssues/LinkedData.html.

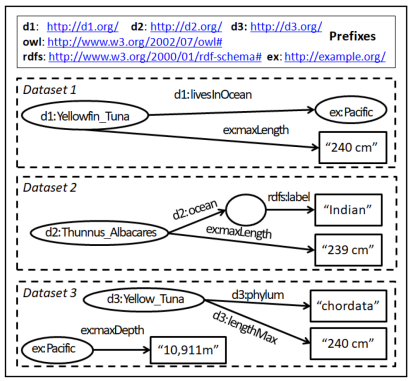

Fig. 1. Example of Linked Data with 9 triples.

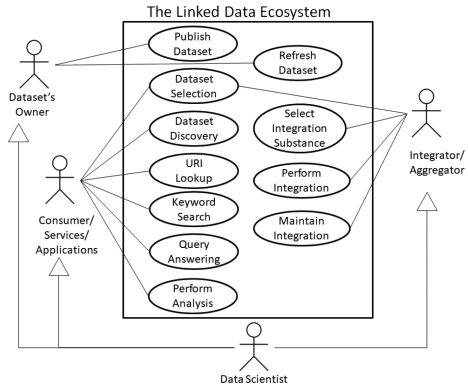

Fig. 2. Use case diagram for the Linked Data ecosystem.

2.2 The Linked Data Ecosystem

To understand the world of Linked Data and the involved stakeholders, we could consider them as a single *ecosystem*. Figure 2 provides the Use Case Diagram, and below we identify the main actors and use cases of that ecosystem.

Dataset's Owner. The owner of a dataset can be a public organization (e.g., a municipality, a research centre, a university), a private sector organization (e.g., a company, like BBC, NGOs, etc.) or even an individual that owns and is responsible for creating, keeping, maintaining, and publishing the dataset.

Consumer, Services, or Applications. This actor corresponds to entities, services or applications that consume data for various reasons, i.e., for providing services of various levels or granularity like *Dataset Discovery*, *Dataset Selection*, *URI Lookup*, *Keyword Search*, and *Query Answering* services. For instance, *LODLaundromat* [126] offers URI Lookup and Keyword Search for over 650,000 documents. Other applications, such as *Swoogle* [34] and *Watson* [30], offer keyword search services for the Semantic Web datasets and resources, while *LODsyndesis* [105] offers services for *Dataset Discovery* and *Dataset Selection*. Finally, this actor includes end-user applications (like smart phone applications) that exploit Linked Data for supporting their functionality.

Integrator/Aggregator. This actor captures individuals or organizations whose objective is to integrate a number of datasets and provide integrated access services. The final output can be exploited by the members of the corresponding community. One example is Europeana Foundation, which is responsible for *Europeana* [76], which is the European digital platform for cultural heritage. This platform combines data from more than 3,000 institutions across Europe while these data were transformed into Linked Data and are represented in the Europeana Data Model [76]. In other occasions, international projects play this role. Note that the use case "Perform Integration" is actually the process that will be analyzed in this survey.

Data Scientist. A Data Scientist can be considered as a special case of Consumer and Aggregator/Integrator. A data scientist usually has to find and select the appropriate datasets (or subdatasets) for his/her needs and may have to aggregate and integrate data to perform the intended analysis. Moreover at the end, the data scientist can publish the results of this analysis as a new dataset (therefore it could be considered as dataset owner).

2.3 Related Surveys

Several surveys have been published in the database world about *Data Integration*, i.e., surveys assuming the relational data model (such as Reference [65]), surveys assuming the semi-structured

data model (e.g., Reference [11] includes approaches for integrating XML databases), surveys that concern big data integration for both models (e.g., Reference [41]), as well as surveys for semantic integration focusing on "ontology-based" approaches [82, 113]. Concerning Linked Data, there are surveys for various individual tasks of the general integration process. Specifically, Reference [74] surveys techniques and approaches for scalable *distributed RDF processing, querying and reasoning*, e.g., search engines, federated query systems, rule-based reasoning, and so forth. Reference [144] provides a literature review for the field of *ontology matching* for the decade 2003–2013, whereas the authors in Reference [108] compare the features of various tools and frameworks that perform *instance matching*. In Reference [1], the objective was to survey how the "Semantic Web technologies can aid in data discovery, acquisition, integration, and analytical querying of external data, and thus serve as a foundation for exploratory on-line analytical processing (OLAP)." The authors analyzed the steps that should be carried out for creating data warehouses and answering *OLAP queries*. Moreover, Reference [130] compares novel *SPARQL endpoint federation engines* in many dimensions and details the tools that have been created for this specific method of integration, while Reference [81] surveys approaches that support scalable *distributed SPARQL query evaluation*. Reference [123] highlights the strong need for holistic data integration approaches that can integrate many data sources (and not be limited only to pairwise matching). Reference [29] surveys approaches for *visualizing* Linked Data, Reference [10] surveys a number of systems for *data visualization* and exploration, while Reference [153] surveys methods for supporting *faceted exploration* over RDF datasets. Concerning the *quality* of Linked Data, the authors in Reference [168] survey 21 approaches and describe 26 data quality dimensions, e.g., *accuracy*, *interlinking*, *conciseness*, *consistency* and others, while they introduce corresponding metrics and approaches for each dimension. Moreover, Reference [119] surveys approaches focusing on *knowledge graph refinement* (mainly on error detection and data completion), which are of primary importance for improving the quality of a single dataset (that could be an integrated dataset). Finally, *OCLC Research Team* has provided a survey for hundreds of projects and services from the domain of *digital libraries* that exploit Linked Data principles [145]. The key difference between our survey and the aforementioned ones is that they focus on various individual tasks of the general integration process, whereas we emphasize on the whole integration process and methods that can be applied for large number of datasets (e.g., thousands of datasets).

3 WHY DATA INTEGRATION IS DIFFICULT

Information integration aims at offering unified access services over a set of information from heterogeneous datasets (structured, semi-structured or unstructured), which can have different conceptual, contextual, and typographical representations. Integration is not easy for various reasons. Most of these reasons has attracted the interest of database community for decades, e.g., for relational model [146, 147] and in the area of *Semantic Integration* [82, 113]. Below, we list six main reasons, each exemplified using the running example of Figure 3, which shows the three sources of Figure 1, i.e., D_1, D_2, and D_3, in the context of four integration scenarios.

(a) **Different Authorities**. The datasets are produced, kept or managed by different organizations in different formats, schemas, models [82, 113], locations, systems and licenses. There is not any "centralized control system," therefore, each publisher decides how to produce, manage and publish a dataset based on its needs and choices. For instance, in the marine domain the dataset of *Fishbase* (http://www.fishbase.org), that contains information about the "taxonomy, geographical distribution, biometrics, population, genetic data and many more" [104], is accessible through an SQL server, therefore one needs a fetcher to download it and a transformer to RDF, for linking it with other datasets. On the contrary, for *Ecoscope* dataset (http://ecoscopebc.mpl.ird.fr/joseki/ecoscope), which contains "geographical data, pictures and information about

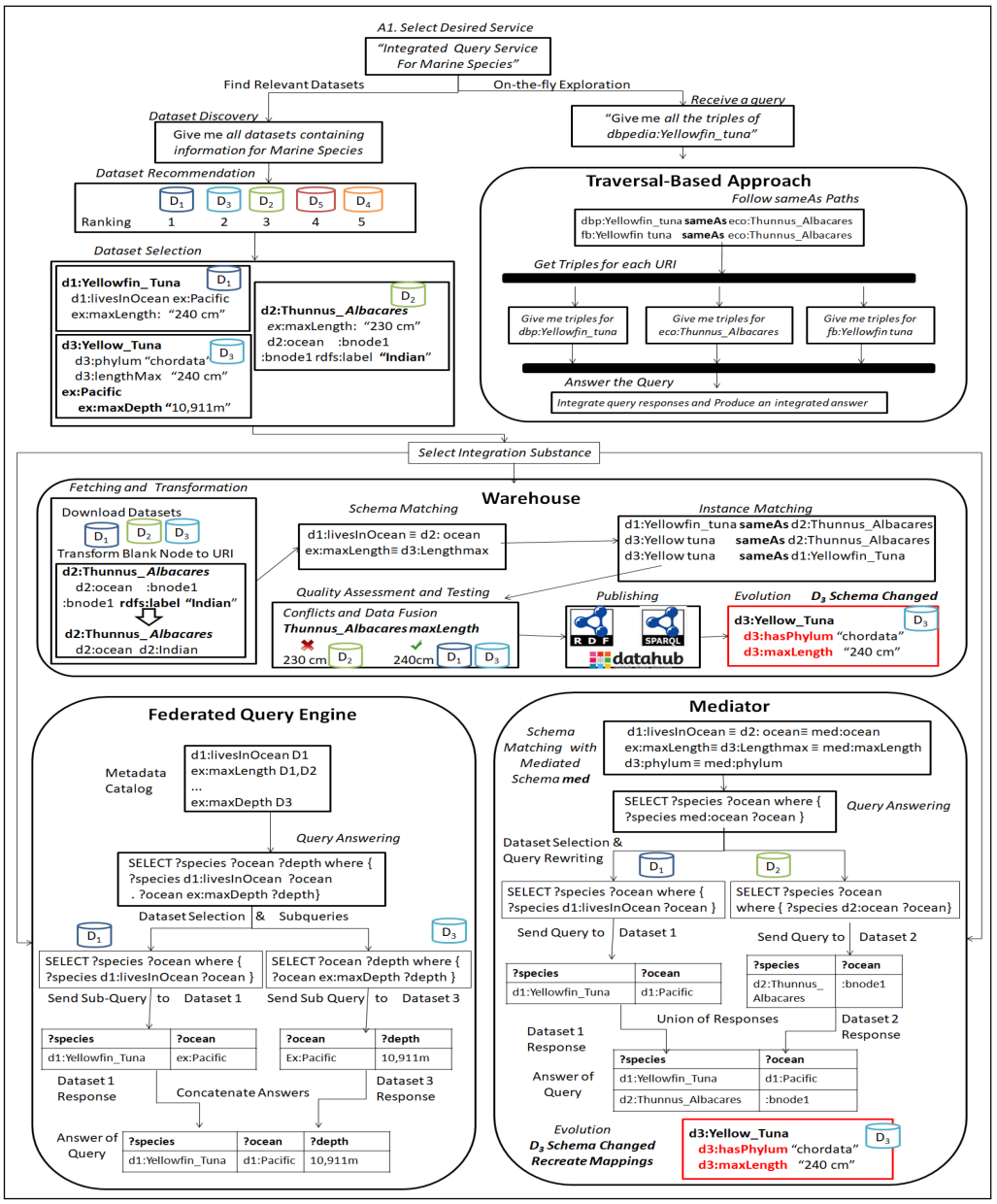

Fig. 3. Running example containing the steps of four different integration substances.

marine ecosystems" [104], transformations are not required since this dataset has already been published in RDF format.

(b) **Naming.** The same real-world entities or relationships are referred with different URIs and names and in different natural languages, while natural languages have synonyms and homonyms that make harder that automatic connection. For instance, the URI of the species *Thunnus Albacares* in DBpedia is http://www.dbpedia.com/Thunnus_Albacares, while in Ecoscope the corresponding

URI is http://www.ecoscope.com/thunnus_albacares. Moreover, the aforementioned species has 348 common names in 82 different natural languages [151]. In addition, we often have to tackle the problem of homonyms, since the same name can describe two or more different real-world entities. For example, "Argentina" is used to refer to the country (http://dbpedia.org/resource/Argentina) but also to a fish genus (http://dbpedia.org/resource/Argentina_(fish)).

(c) **Complementarity**. The datasets usually contain complementary information, e.g., consider two datasets about the same domain each modeling a different aspect of the domain. The commonalities between these datasets can be very few and this does not aid automated linking and integration. For example, a dataset can include data about the predators of marine species, and another one can contain data about the selling price of a species in a fish market. In the former, the species can be referred by its scientific name, while in the latter by its commercial code.

(d) **Errors, Conflicts.** The datasets can contain data that are erroneous, out-of-date or conflicting. For example, in the marine domain *Fishbase* mentions that the max length of *thunnus albacares* is 239cm while Wikipedia (whose content is used from many RDF sources such as DBpedia) states that its max length is 240cm, meaning that conflicts can occur even because one dataset is more precise than another. Other conflicts are due to erroneous information, e.g., suppose that one dataset states that the capital of Australia is Sydney and several other datasets that is Canberra. Finally, out-of-date data are very common in many domains, e.g., the current team of a football player (this is, however, related also to difficulty (f) described below).

(e) **Different Conceptualizations.** The datasets about the same domain may follow different conceptualizations (or modeling decisions) of the domain [82, 113], i.e., they have different schemas (as we would say in the relational database world). For instance, some datasets conceptualize an address by using one property (e.g., ⟨:Michael, :hasAddress, "Street 1, Heraklion, 71303"⟩, others use one property per address part (e.g., ⟨:Michael, :street,"Street 1"⟩, ⟨:Michael, :city, "Heraklion"⟩ and ⟨:Michael, :postCode, "71303"⟩). Moreover, other datasets use a blank node for representing an address. Consequently, there is not a general pattern that the creators of the datasets follow for representing the information for a specific domain.

(f) **Evolution.** Everything changes: the world, the ontologies (e.g., see Reference [55] for a survey of ontology change), the data. Thereby, integration actions that have taken place may have to be updated or revised. This is also a source of possible conflicts as stated earlier.

4 THE DATA INTEGRATION LANDSCAPE

The data integration landscape is wide and complex. For approaching it in a structured manner, we can describe an integration process through a multidimensional space. Such a space should allow describing each integration method as one or more points of the multidimensional space in a clear manner. Specifically, we introduce the space defined by the cartesian product of *five dimensions*

$$\begin{aligned} IntegrationLandScape \quad = \quad & (DatasetTypes \times BasicServicesToDeliver \times IntegrationSubstance \\ & \times InternalServices \times AuxiliaryServices) \end{aligned}$$

Figure 4 illustrates the basic values that correspond to each dimension. Below, we briefly describe each one, while a detailed description for each one is given in the next sections.

- *DatasetTypes* (or input types) refers to the different dataset types that can be used as input, e.g., RDF files, relational databases, HTML with embedded annotations (e.g., RDFa, JSON-LD). The *dataset's owner* (recall the Linked Data Ecosystem in Section 2.2) is responsible for the type of the dataset.

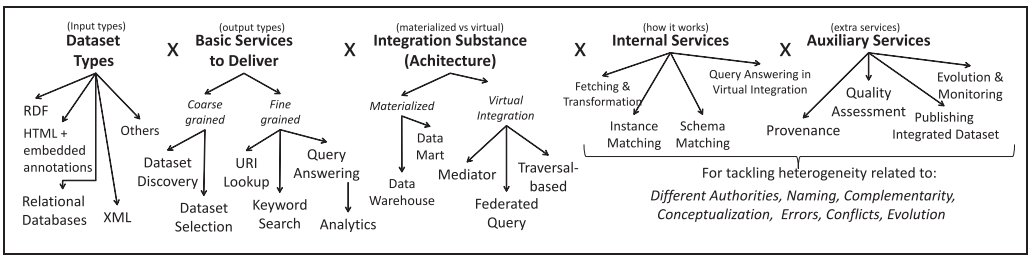

Fig. 4. The dimensions of data integration landscape.

- *BasicServicesToDeliver* (or output types) refers to the main purpose of integration, i.e., what services we would like to offer by integrating several datasets, e.g., see those shown in Figure 4. This dimension corresponds to *Consumer, Services or Applications* actor of Section 2.2.
- *IntegrationSubstance* (or *Integration Architecture*) refers to the different integration substances, physical (materialized) versus virtual, and their specializations (e.g., see those shown in Figure 4). The responsible actor (according to Section 2.2) is the *integrator/aggregator.*
- *InternalServices* (i.e., how it works) refers to the services used during the integration process, e.g., transformations, bindings, and so on, to "connect" the pieces of data, thereby, the responsible actor is the *integrator/aggegator* (see Section 2.2).
- *AuxiliaryServices* (i.e., extra output types, beyond the core ones) refers to services that can be optionally exploited/offered either before or after the integration process, related to provenance, evolution, quality and others (again *integrator/aggregator* is the responsible actor).

For tackling the various discrepancies (as mentioned in Section 3), which is actually the "duty" of the *InternalServices* mentioned before, the various integration approaches essentially attempt to "connect" the data of the underlying datasets. Generally, datasets can be connected (or linked) through (a) instance links, (b) schema concepts, and (c) constants, i.e., literals. According to Reference [106], LOD cloud datasets are mainly linked through schema concepts (99% of datasets' pairs share RDF schema elements) and literals (78% of datasets' pairs share literals), while only 11.3% of datasets' pairs contain common instances. One kind of connection is through *canonicalization*, e.g., an integration system can decide to transform every occurrence of "UK" and "Great Britain" to "United Kingdom." In this way, semantically equivalent elements can get the same single representation. Another kind of connection is through *binding*, i.e., by adding extra relationships (data in general) that connect these elements, e.g., "UK" ≡ "Great Britain." We use the term "binding" to refer to what is called *correspondence, mapping, link*, and so on. We can distinguish the following cases based on the **semantics of these bindings**; in some parts of the discussion below, we shall use examples from the two small datasets shown in Figure 5:

- *Taxonomy-based relations*
  - —*Exact-ones.* Here, we have relations expressing equivalence, e.g., owl:EquivalentProperty, owl:sameAs, skos:exactMatch, or difference, e.g., owl:DifferentFrom.
  - —*Inexact-ones.* Here, we have connections between elements of different granularities, e.g., ⟨Researcher, subclassOf, Person⟩ and ⟨livesAt, subpropertyOf, staysAt⟩, connections between different accuracy levels, e.g., 84 vs. 84.9, or connections between similar concepts by using relations such as skos:closeMatch.

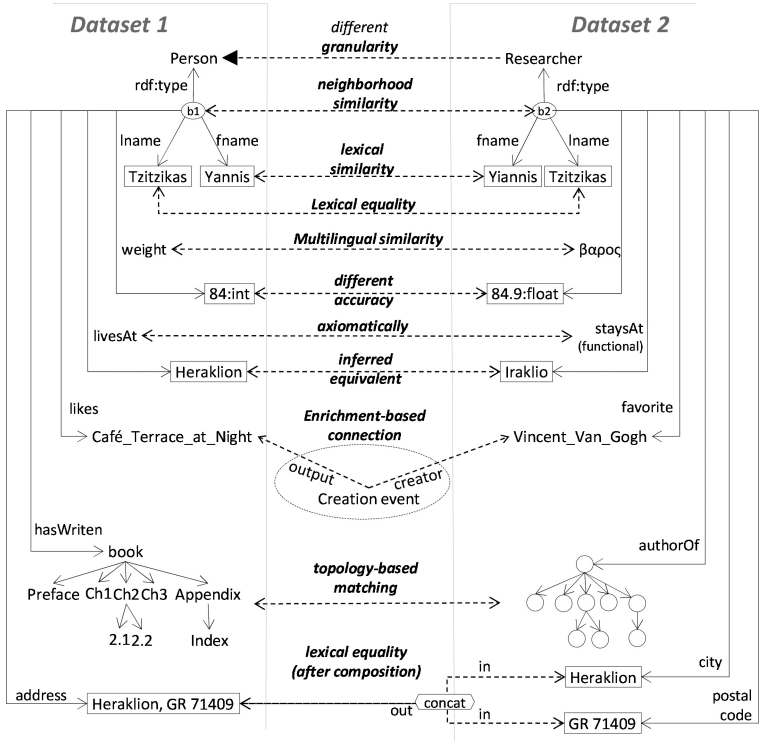

Fig. 5. Example of two datasets and possible bindings.

- *Enrichment-based*. Here the binding of two elements is achieved through non-taxonomic properties, or paths of properties, e.g., see how Cafe_Terrace_at_Night of Dataset1 can be connected with Vincent_Van_Gogh of Dataset2.

Now there are several **types or sources of evidence** for creating bindings of the aforementioned kinds:

- *Axiomatically* by users (designers, curators), e.g., the triple ⟨leavesAt, subpropertyOf, staysAt⟩ can be provided by a designer.
- *Name similarity* (string similarity, phonetic similarity, stemming), e.g., Yannis ~ Yiannis can be detected through EditDistance.
- *Neighborhood (direct or indirect) similarity*, e.g., b1 ~ b2 can be detected by blank node matching techniques.
- *Natural Language Processing-based*, e.g., entity or relation mining over the literals associated through rdfs:label with a URI can be used for connecting that URI with the URI of the identified (in the literal) entity(-ies). This type of evidence can be based on lexicons, thesaurus, translators, e.g., multilingual similarities, such as weight ~ $\beta\alpha\rho o\varsigma$ (i.e., weight in greek language) in Figure 5, can be detected by using a translator, and word embeddings (e.g., word2vec [97]), where words having similar meaning are represented in a similar way.
- *Common Instances*, for example two classes can be connected with a taxonomic relation based on the fact that they have common instances (through the ostensive method, or through inductive and machine learning-based methods).

- *Inference-based* (or *Reasoning-based*), e.g., the equivalence Heraklion ≡ Iraklio is inferred, because the staysAt is a functional property and livesAt is a subproperty of staysAt.
- *Topology-based similarity*, e.g., all nodes of the book in the left side can be matched with the blank nodes subgraph in the right side (because these two graphs are isomorphic).
- *Usage-based*, e.g., frequently accessed together entities can be connected because of this. For example, Yannis could be connected with Γιαννης (which is the same name written in Greek characters) if these two words frequently co-occur in the log file of a search system.

We observe that to integrate properly these two very small and simple datasets, that contain information about the same (very narrow) domain, we have to combine various kinds of evidence and create various types of binding.

5 SURVEYING THE INTEGRATION METHODS

For surveying the various integration methods for Linked Data, apart from studying the literature [2008–2017] (the main works, since it is impossible, for reasons of space, to include the entire literature), we considered also our experience from various EU projects. This section is structured according to the dimensions (or aspects) of the *Data Integration Landscape* introduced in Section 4. For each dimension, we describe its role, the related challenges, and the related methods, techniques and tools. Specifically, in Section 5.1, we discuss the different dataset types, in Section 5.2, we describe the basic services that can be delivered through an integration process, in Section 5.3, we focus on the integration substance, while in Section 5.4, we describe the internal services and in Section 5.5 the auxiliary services. Finally, in Section 5.6, we classify 18 integration tools according to the dimensions in Section 4.

5.1 Dataset Types

DatasetTypes refers to the different dataset types that can be used in an integration process as input. In this survey, we focus on datasets represented in RDF format. However, we should note that there are several ways and tools for mapping relational databases to RDF format [141] and for converting CSV (comma separated values) files to RDF [46]. Moreover, a huge volume of RDF data can be extracted through HTML web pages [18, 121]. Specifically, billions of web pages contain semantic annotations by using *Microformats*, *RDFa*, and *HTML Microdata* mainly inside their <*body*> element, and *JSON-LD* scripts predominantly inside their <*head*> section.

5.2 Basic Services To Deliver

One aspect of primary importance is what is the *purpose* of integration, i.e., why we want to integrate data, what we want to achieve and to eventually deliver in terms of input and desired output. Some forms of the desired integrated access can be simple, while other forms can be more complex. To this end, below we dichotomize such services to *Fine-grained* and *Coarse-grained* services:

- *Fine-grained (FG) Services.* Here the objective is to find, select and assemble "pieces" of data. To this category of services, we can distinguish three different levels: *Level I. Global URI Lookup Service* (analyzed in Section 5.2.1), *Level II. Global Keyword Search* (analyzed in Section 5.2.2) and *Level III. Integrated Query Answering Service* (analyzed in Section 5.2.3).
- *Coarse-grained (CG) Services.* Here the objective is to find or select *entire datasets*. In this category of services, we have: *Dataset Discovery & Selection* (analyzed in Section 5.2.4).

5.2.1 FG: Level I. Global URI Lookup Service. This kind of service can be used for finding all the URI containing a substring or all the equivalent (or similar) URIs of a given URI u. For realizing such a service, we have to tackle difficulty (b), i.e., the problem of synonyms and homonyms, which in turn requires the execution of various tasks including:

- *Cross-dataset completion* of the `owl:sameAs` relationships for completing (with respect to symmetry and transitivity) the relationships that are already recorded, otherwise the response would be incomplete. The same is true for schema elements, e.g., for `owl:equivalentClass`.
- *Matching methods* for individuals or schema elements (instance matching is described in Section 5.4.3 while schema matching is described in Section 5.4.2) for identifying new equivalences between URIs that have not been recorded. However, we should note that not all entities (or concepts) have necessarily URIs, in the sense that some of them can be represented as literals, or even not modeled in the datasets themselves (they could be called hidden intermediate concepts or entities [38]).

5.2.2 FG: Level II. Global Keyword Search. This kind of service can be used for finding URIs, triples, literals and datasets relevant to an information need that has been expressed as a keyword query. Such services can be based on Information Retrieval techniques, semantic-based techniques, or both. The responses of such services is a list of elements ranked according to their estimated relevance to the user query. It is not hard to see that having achieved an effective (of good quality) I-service, certainly aids the provision of II-services, since without complete URI lookup the II-services will miss relevant elements and thus will have low recall.

5.2.3 FG: Level III. Integrated Query Answering Service. This kind of service answers complex queries containing data derived from more than one dataset over any integrated system. For instance, such a service should be able to answer the following query: "Find the ecosystems, waterareas and countries that http://www.dbpedia.com/Thunnus_Albacares is native to, and the common names that are used for this species in each country, as well as their commercial codes." One difference between level III and level II services is that a III-service takes as input a query expressed in a structured query language (SPARQL), while II-services receive as input queries expressed in natural language. Consequently, one major difference between III-services and I/II-services is that a III-service presupposes a common conceptualization that allows formulating the query and it also presupposes the existence of data according to this conceptualization (for evaluating the query). The latter is not always feasible, since we may miss the needed datasets mainly due to the different conceptualizations or the complementarity of datasets (difficulties c and e), i.e., we may not have enough common data to establish connections. Another difference between III-services and I/II-services is that it is much more difficult to achieve a *quality response* to a complex query. For example, suppose that an I-service has recall 80% (e.g., that it only finds 80% of the URIs or triples about a real-world entity). Now a query that contains a condition that involves 2 URIs (e.g., all x such that (u_1, X, u_2) is expected to have recall equal to 64% (=0.80 * 0.80), with k URIs, the recall would be 0.8^k. Clearly, the more complex the query becomes, the harder to achieve good quality.

5.2.4 CG: Dataset Discovery and Selection. It refers to the discovery of the most relevant datasets to a given keyword, dataset, or URI and to the selection of the most desired datasets that fulfill the requirements that the selected integrated access system is intended to serve.

Context. If there is prior knowledge for the datasets that will be integrated (e.g., in MarineTLO warehouse [151], then the authors already knew which datasets will be used for constructing the warehouse) there is no need to exploit such a *Dataset Discovery* service. On the contrary, one can exploit such a service if there is no knowledge about what datasets exist, or there is prior knowledge for some datasets that will be surely used, however, more datasets are needed. For instance, Figure 3 shows an example where the scientist (or user) desires to find datasets whose domain is about *Marine Species* by using a keyword search engine that returns a ranked list of datasets.

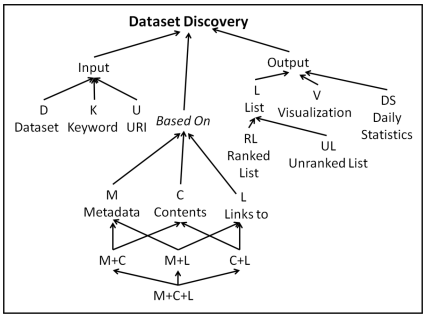

Fig. 6. The different criteria of dataset discovery.

Table 1. Categorizing Existing Dataset Discovery Approaches According to the Criteria of Figure 6

Approach	Input	Based on	Output
Nikolov et al. [112]	D, K	C	RL
Leme et al. [91]	D	M	RL
Wagner et al. [163]	D, K	M+C	RL, V
Ellefi et al. [44]	D	M	RL
LinkLion [109]	D	M	UL
RapidMiner Link [127]	U	L	UL
LODVader [6]	D	M	RL, V
LODsyndesis [105]	D, U	M+C	UL, V
Lotus [75]	K	C	UL
Swoogle [34]	K, U	M+C	RL
Sindice [116]	K, U	M+C	RL
SWSE [72]	K, U	M+C	RL
Watson [30]	K, U	M+C	RL
Datahub.io	K	M	UL
Google Dataset Search [18]	K	M	RL
SpEnD [166]	K	M	UL, DS, V
SPARQLES [159]	K	M	UL, DS, V

Difficulties. The heterogeneity in terms of format, schema, and so on, the existence of different URIs referring to the same real-world entity, and datasets evolution, i.e., difficulties (a), (b), and (f), makes it difficult to discover valuable and relevant datasets. Additionally, difficulties (c), i.e., complementarity of information, and (e), i.e., different conceptualizations, can complicate that process.

Categorization. Figure 6 depicts the different criteria that can be employed for characterizing the *Dataset Discovery* approaches. One criterion is how the information need is expressed: as a *Keyword-query*, as a *URI* or as a *Dataset*. Another criterion is the method that is used for discovering relevant datasets: there exist *Metadata-based* approaches that depend on metadata such as measurements, statistics, and dataset descriptions, *Content-based* approaches, which exploit the contents of the datasets for discovering relevant datasets, and *Links-to* approaches that discover relevant URIs and datasets on-the-fly by traversing equivalent links. Regarding the output, the simplest one is an *Unranked List*, which just returns a list of datasets (or links with their provenance) without ranking, while the output can be a *Ranked List*, where a score is assigned for each dataset, e.g., by exploiting *Dataset Recommendation* techniques. Finally, a *Dataset Visualization* output can be more informative for the human users for helping them to understand and explore the structure and the interconnections of Linked Data and lead to an efficient and intuitive interaction with them, while *Daily Statistics* (as an output) are important for checking the evolution of each dataset.

Approaches. Table 1 lists a number of approaches and categorizes them according to the dimensions of Figure 6. Below, we describe in brief these approaches.

Dataset-based Approaches. Leme et al. [91] proposed a probabilistic classifier based on Bayesian theory, for identifying the most relevant datasets that can be interlinked with a given dataset, i.e., they create a ranking with respect to the probability of a dataset to be relevant with a given one. *LinkLion* [109] is a service collecting pairwise mappings for 476 RDF datasets, while one

can discover mappings for a given dataset. In Reference [44], it is presented a dataset recommendation approach for detecting potential datasets, that can be linked to a given dataset and relies on the common schema elements among the datasets. *LODVader* [6] uses Bloom filters to compare, index and count links between RDF distributions in the streaming process. One can upload a dataset description file and that system compares its metadata with the existing datasets' metadata. Then, it provides to the users a LOD Diagram showing the discovered links for their datasets. Nikolov et al. [112] proposed a method that takes as input a single dataset and uses an index and keyword search for retrieving the most relevant datasets to the given one. *LODsyndesis* [105–107] offers content-based measurements for 400 datasets, e.g., one can find the "K most connected datasets to a given one."

Keyword-based Approaches. *Google Dataset Search* engine [18] crawls and indexes metadata from thousands of datasets and HTML web pages. By using that engine, one can retrieve the most relevant datasets to a set of keywords. Moreover, *Datahub.io* offers also a keyword search mechanism by exploiting the metadata of datasets that have been uploaded from several organizations and users. *Semantic Web Search Engine* (SWSE) [72] resembles a classical search engine, i.e., it crawls and indexes RDF data and provides a keyword search for easing the search, exploration and retrieval of RDF data. *Swoogle* [34] crawls and retrieves several semantic web documents by exploiting metadata and by traversing links. All the retrieved documents are indexed and analyzed by using several metrics for computing ranks (e.g., ranks for ontologies). The users can exploit the keyword search functionality for retrieving results about these documents. *Sindice* [116] uses also a crawler for discovering and fetching RDF files, while it connects to SPARQL endpoints for retrieving RDF datasets. It uses several indexes (e.g., for URIs and literals), while one can submit a keyword query in this system for finding relevant data. *Watson* [30] offers advanced functionality for searching semantic web resources, i.e., one can search in documents and ontologies, find metadata, explore ontologies, write their one SPARQL queries and use several metrics by selecting their own filters. In Reference [163], recommendations are provided through keyword search for integrating identical schema and instance elements of different datasets. The users can discover more sources, that are connected with the selected ones by exploiting measurements that try to find similarities between the datasets. *Lotus* [75] is a text engine returning URIs and their provenance for a given set of keywords. Finally, *SPARQLES* [159] and *SpEnD* [166] contain a catalog of SPARQL endpoints, however, their main objective is to monitor the evolution of SPARQL endpoints over specific time periods by providing daily statistics (and visualizations).

URI-based Approaches. The URI-based approaches are based either on indexes or on the existence of dereferencing HTTP URIs, where one can discover relevant URIs on-the-fly (by traversing `owl:sameAs` or other kinds of relationships). *LODsyndesis* [105, 107] provides a global entity lookup service based on a `owl:sameAs` catalog, which retrieves all the datasets (and triples) where a URI u (or an equivalent URI of u) exists. *RapidMiner Link Explorer* [127] takes as input a URI and discovers on-the-fly relevant URIs and datasets by following `owl:sameAs` paths. The advantage of on-the-fly approaches is that one can explore a large number of URIs for the same entity, while by using an index, the number of URIs for the same entity is stable. However, they strongly depend on dereferencing HTTP URIs, since a path terminates when a non-dereferencing URI is visited. Finally, a number of keyword searching services [30, 34, 72, 116] also offer a URI lookup service.

5.3 Integration Substance (Materialization vs. Virtualization)

Here, we describe the *Integration Substance* (or *Integration Architecture*), which can be *materialized* or *virtual*. The corresponding integration approaches are described below and Figure 7 shows their main steps.

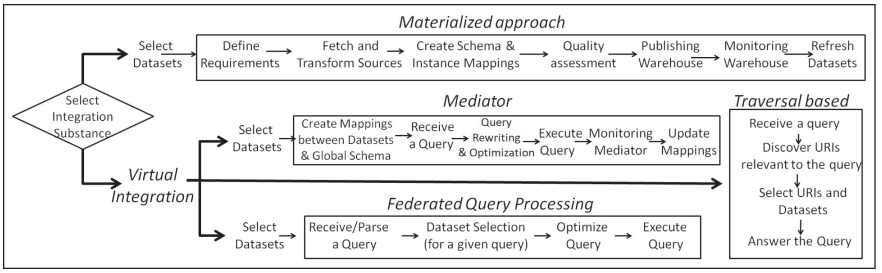

Fig. 7. Steps for materialized and virtual integration.

Materialized Approach. In the materialized (or warehouse) integration, the integrated data are stored in a single repository [21, 151]. As a first step (see upper part of Figure 7), one should select the datasets (or views of specific datasets) that are appropriate for fulfilling the requirements of the warehouse, which are defined in its "design" phase. In the "creation phase," it is a prerequisite to download (or fetch) the underlying datasets and usually to transform them to a target model before uploading them in that repository, i.e., since datasets use different schemas, formats and so forth. Moreover, mappings among the underlying datasets in both schema and instance level should be defined, for enabling the answering of complex queries, that combine information from two or more datasets. A crucial step is the "testing phase," where one assess the quality of the constructed repository, by using several metrics and by taking into consideration the defined requirements. In the "monitoring phase," one should monitor the underlying datasets for identifying possible changes in one or more of them. Such a change can result to the reconstruction of the whole warehouse, i.e., "refresh phase." Regarding the benefits of this integration substance, (a) it is more flexible in terms of transformations, (b) the stability and robustness of the materialized repository do not rely on the datasets' servers, i.e., one should access such a server only either for fetching a dataset or for monitoring possible changes, instead of answering queries, and (c) it can offer faster responses, mainly in query evaluation, and secondarily in several other tasks, e.g., for applying techniques for instance or schema matching. Concerning the drawbacks, there is a cost for hosting such a repository, while it needs a periodical monitoring and refresh. Figure 3 contains an example with the required steps for building a *Semantic Warehouse*. Moreover, we should mention OLAP approaches, which is a special case of materialized data integration. In this case, data are described by using a star-schema (modeling entities of a single type), while "data are organized in cubes (or hypercubes), which are defined over a multidimensional space, consisting of several dimensions" [160]. That technology is mainly used by enterprises, for producing critical analytics (aggregate queries), by using internal data (e.g., sales of a company), or/and external data, e.g., through the exploitation of semantic web technologies [1]. OLAP queries mainly belong to *Analytics* service of *BasicServicesToDeliver* dimension (see Figure 4). In Reference [1], several OLAP approaches are surveyed, while "*The RDF Data Cube Vocabulary*" (http://www.w3.org/TR/vocab-data-cube) can be exploited for publishing multi-dimensional data (results of aggregated queries) in RDF format. Finally, we should also mention *Data Marts*, which are "small units of a Data Warehouse that are dedicated to the study (analysis) of a specific problem" [17]; i.e., they belong to *Analytics* service of *BasicServicesToDeliver* dimension (see Figure 4).

Mediator (Virtual Integration). In the mediator approach, the data remains in the original sources [21, 151], while sources can be unaware that they are part of an integration system [74]. Concerning its "'design phase" (see central part of Figure 7), one should select the datasets that will be used and to define a mediated schema, which is essential for supporting query translation among the different models of the underlying datasets' schemas. Thereby, the mappings between

the mediated schema and each dataset should be created. The core functionality of a mediator contains three main steps. Initially, a query, expressed by exploiting the mediated schema, is received. The query is disassembled in smaller sub-queries, where the mappings are exploited for performing *Query Rewriting*, i.e., as stated in Reference [22], "the problem of query rewriting consists in reformulating the query into a (possibly) equivalent expression, called rewriting, that refers only to the source structures." Therefore, such a process makes it feasible each sub-query to be answered by a specific dataset, and for optimizing the query execution plan. Concerning the last step, each sub-query is sent to a specific dataset's server, which in turn sends a response with the answer of such a sub-query, and the responses of all the sub-queries are merged for providing to the user the answer of the initial query. Concerning the "monitoring phase," it is of primary importance to monitor the underlying sources for detecting possible changes that can result to the reconstruction of the mappings between the datasets and the mediated schema. Regarding the advantages of a mediator, there is no need to pay a cost for hosting the dataset, while it can access in real-time updates of datasets' content. On the contrary, its efficiency, quality and complexity relies mainly on the sources' servers. In Figure 3, we can see an example for a *Mediator* approach.

Federated Query Processing (Virtual Integration). "It refers to the process of running SPARQL queries over several SPARQL endpoints" [120]. The underlying sources either use terms from the same schemas for describing their data, or they use common URIs for describing specific entities. As a consequence, it is a prerequisite that specific schemas and URIs are reused for achieving federated query processing, i.e., for performing joins among two or more sources. As it can be seen in Figure 7, the first step is to select which datasets will be used for answering queries. Afterwards, the federated query processing contains the following steps: "*Query Parsing*, *Data Source Selection*, *Query Optimization*, and *Query Execution*." *Query Parsing* is the process of parsing and transforming a given query expressed by using SPARQL query language into a query execution tree [115], while *Data Source Selection* is used for finding the relevant datasets (i.e., SPARQL endpoints) for a triple (or a set of triples) pattern of a given SPARQL query [62]. By having selected the datasets for a given query, *Query Optimization* process starts for placing the triple patterns into groups, and it is used for determining in an efficient way the order of joins and triple patterns. The last step, i.e., *Query Execution*, is performed for answering the initial query. Similarly to a mediator approach, the data remains in the original sources. However, a federated query approach does not depend on a global schema (and thus there is no need to create mappings among different sources); instead, it assumes that the underlying sources describe their data by using a common data model (therefore, it is not capable to tackle heterogeneities such as different conceptualization). Moreover, sources can be aware that they participate in such a federation system [115]. In Figure 3, we can observe an example of a federated query engine.

Traversal-based Integration (Virtual Integration). The traversal-based integration depends on the live exploration of data links at query execution time [68]. First, such an approach receives a query and searches for URIs given either in the query body or as additional parameters. Second, it discovers URIs that are relevant to that query by traversing paths (e.g., `owl:sameAs` paths), for finding more URIs that can be possibly exploited for enriching the results of the given query. Afterwards, the URIs and datasets that will be used for answering the query are selected, since it is not necessary to use all the relevant URIs (such a selection can be assisted by predefined rules). Finally, it collects the answers for each URI and returns the final answer (containing data from one or more datasets) to the user. One major advantage of this approach is the live exploration and discovery of relevant datasets that were unknown, since it does not need prior knowledge for answering a query. Consequently, there is no need to store and transform data in such systems, i.e., data remains in the original sources. On the contrary, the data access time can be a drawback due to the recursive process that is usually followed, while the existence of few available deferencable

links makes it difficult to discover relevant data. Moreover, since datasets are explored at real time, it is difficult to measure their quality, while the possible chain of links can be huge. In Figure 3, we can see that a smaller number of steps is required for a *Traversal-based* approach. Comparing to other Virtual Integration approaches, it neither uses a global schema (like in a mediator approach), nor it knows *a priori* all the candidate sources that can be used for answering a query.

Hybrid Integration. A *Hybrid Integration* approach can share characteristics from two or more integration substances, e.g., suppose an approach, where a part of data are fetched and transformed, while an other part of data is explored at query time (e.g., by following `owl:sameAs` paths).

5.4 Internal Services

This set of services are usually performed during the integration process. In Section 5.4.1, we discuss *Fetching and Transformation*, in Sections 5.4.2 and 5.4.3, we discuss *Schema* and *Instance Matching*, respectively, whereas in Section 5.4.4, we describe the process of *Query Answering* in virtual integration.

5.4.1 Fetching and Transformation. It aims at fetching the data that will be used in a materialized approach and at transforming them into a common format.

Context. It can be applied in a Materialized approach, since in a Virtual approach, data are left in their original sources. We can see in Figure 3 (in the materialized approach) that the scientist/user downloaded the datasets and transformed the blank nodes of D_2 to URIs.

Difficulties. It can tackle difficulties (a) and (e), i.e., the heterogeneity of datasets in terms of format, schema, and modeling decisions that datasets follow for the same real-world objects.

Categorization. A variety of access methods can be offered from each dataset for *fetching* its contents, (all contents or a specific "slice" of a dataset). In particular, for many RDF datasets, a SPARQL endpoint or an RDF dump is provided, while alternative access methods include accessible files through HTTP, a JDBC connection and others. The datasets that are not provided in RDF format need to be transformed to that format, i.e., *format* transformation. For some datasets a *logical* transformation should be also performed, i.e., for enabling the representation of these datasets' content by using a core ontology. The *logical* transformation can contain rules that change the language of a literal, rules that transform the type of an RDF resource, e.g., transform a URI to a blank node, a URI to a literal, a blank node to a URI, and so on, or rules for fixing syntactical errors.

Approaches. Concerning *Materialized* approaches, *LDIF* [139] can import data, that are represented in different formats (e.g., RDF/XML, turtle), from SPARQL endpoints or/and by using a crawler. It uses the *R2R* Framework [13] for performing complex *logical* transformations, to integrate data represented through different ontologies into a single one (i.e., conceptualization issues). *MatWare* [156] uses plugins for fetching data that are represented in different formats (e.g., RDF, JDBC, HTTP), and for providing *format transformation* for the fetched data. It also supports *logical transformation* through SPARQL queries and plugins, while it uses the *X3ML* framework [98], which handles in a state-of-the-art way the URI generation and the *logical* transformation. *ODCleanstore* [83] fetches RDF data through a SOAP web service, and executes a defined set of transformers for offering a common data representation (i.e., logical transformation). *KARMA* [84] imports files in many different formats (csv files, RDF, etc.) and transforms all the different data formats into a nested relational data model. It supports also *logical* transformation, i.e., one can combine several ontologies for creating mappings between their data and standard ontologies, while it uses techniques for identifying and suggesting to the user possible mappings. Regarding *Hybrid Integration* approaches, *TopFed* [133] transforms billions of Cancer Genome Atlas (TCGA) data into RDF format, while the transformed sources are described by using the same schema (i.e., format and logical transformation). *RapidMiner LOD Extension* [127] imports data in several formats (e.g.,

csv, excel), while it offers a SPARQL and an RDF Data Cube importer. It supports both *format* and *logical* transformation for representing data in a common format. FuhSen [25] fetches data from different formats (e.g., REST, RDF, SQL) by using wrappers, while data are transformed into RDF molecules, i.e., an RDF molecule is the cluster of all the triples of a specific subject. Furthermore, the authors in [121] fetched billions of HTML pages and they extracted the data expressed in Microdata format and RDFa for producing billions of RDF triples (the produced collection is accessible in http://www.webdatacommons.org/). Moreover, there exists approaches focusing mainly on fetching and transforming datasets. *LODLaundromat* [126] offers a common representation for over 650,000 RDF documents after cleaning them, i.e., they are free of syntactic errors. The authors of Reference [141] have proposed automatic ways for publishing relational databases to RDF, while Reference [142] surveys various approaches for mapping relational databases to RDF, e.g., *CSV2RDF* [46] transforms csv and excel files to RDF.

Evaluation Collections. LODIB [14] is a benchmark for evaluating whether a tool can detect several mapping patterns and perform the required logical transformations (transforming a literal to URI, renaming a class, etc.), for representing the data of three different sources by using a single target vocabulary. Moreover, it measures the performance of each tool in terms of execution time. Finally, in Reference [14] several data transformation benchmarks are mentioned (except for LODIB).

5.4.2 Schema/Ontology Matching (or alignment). It refers to the problem of determining mappings at schema level between schema concepts. i.e., classes (e.g., Person owl:equivalentClass Human) and properties (e.g., staysAt rdfs:subPropertyOf livesAt).

Context. An integration system can (a) use *predefined* (PD) ontology mappings (and possibly their closure) that have been declared (e.g., axiomatically) before the integration process or/and (b) exploit *ontology matching* techniques (OMT) for producing new mappings during the integration process. In the simplest case, when two or more datasets use the same schema (or ontology), there is no need to create mappings between them (e.g., federated query engines depend on datasets that share concepts from same schemas). However, by having two or more datasets with different schemas, mappings are essential for being able to answer queries over the integrated content. By deciding to build a warehouse, one can either (a) create the mappings between each pair of datasets or (b) can define a single global schema and create the mappings between that schema and each dataset, which enables the adequate mapping and integration for data, derived from distinct datasets. By choosing to build a mediator, a (single global) mediated schema is created, whereas mappings between that schema and each dataset's schema are also created. The major advantage of using a single global schema is the less effort that is required for schema mappings: e.g., given K sources instead of having to construct $K * (K - 1)$ pair-wise mappings, only K mappings are required (one for each source). Moreover, the focus is given on one model rather than many. The most commonly used schemas for declaring mappings among ontologies are RDF/RDFS (e.g., rdfs:subClassOf and rdfs:subPropertyOf) and OWL (e.g., owl:equivalentProperty). In Figure 3, for the *Warehouse*, the user selected to create the mappings between the underlying sources, while for the *Mediator*, the mappings were created between a global schema and each underlying dataset.

Difficulties. This process is not trivial, mainly due to difficulties (a), i.e., datasets are usually published by using different and heterogeneous schemas and models, (e) different conceptualization of the same domain, and (f) evolution, i.e., several ontologies are frequently changed. This problem has attracted the interest of the community for decades. For instance, References [146, 147] extensively describe the difficulties of integrating different schemas in the area of relational databases and References [82, 113] analyze the difficulties of matching different ontologies in *Semantic Integration* area.

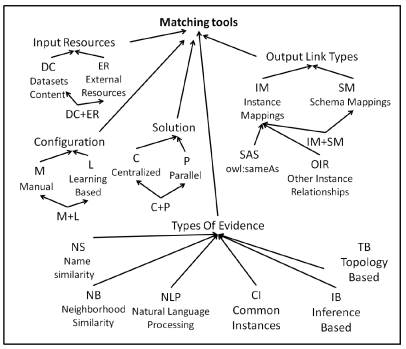

Fig. 8. The different criteria of (schema and instance) matching tools.

Table 2. Categorizing Existing Schema and Instance Matching Approaches for Large-scale Datasets According to the Criteria of Figure 8

Tool	Input Resources	Configuration	Solution	Output Link Types	Types of Evidence
Yam++ [110]	DC+ER	M+L	C	SM	NS, NB, NLP IB
LogMap [78]	DC+ER	M+L	C	SM	NS, NB, NLP, IB, TB
StringOpt [24]	DC+ER	M+L	C	SM	NS, NLP
SBOMT [114]	DC+ER	M+L	C	SM	NS, NB, NLP, CI, IB
Agreement Maker [28]	DC+ER	M+L	C	SM	NS, NB, NLP TB
XMAP++ [37]	DC+ER	M+L	C+P	SM	NS, NB, NLP, IB, TB
WebPie [158]	DC	M	P	SM+IM	IB
PARIS [148]	DC	L	C	SM+IM	NS, IB
Silk [161]	DC	M+L	C+P	SAS+OIR	NS, NLP, TB
LIMES [111]	DC	M+L	C+P	SAS+OIR	NS
LINDA [15]	DC	L	C+P	SAS	NS, NB, IB
MinoanER [42]	DC	L	P	SAS	NS, NLP
CoSum-P [169]	DC	L	C	SAS+OIR	NS, NB, TB
MINTE [26]	DC	L	C	SAS	NB, NS, TB

Categorization. We categorize matching tools (either schema or instance matching tools) in *manually specified linking configuration* tools (i.e., a user manually specifies some rules), *learning-based* tools (e.g., they can exploit machine-learning methods) and tools using *both techniques* [108], as it can be observed in Figure 8. Moreover, since we emphasize on big number of data (and datasets), we can categorize the approaches according to the type of the solution that they offer: a *centralized solution*, which means that the computation is performed in one machine, a *parallel solution*, i.e., a cluster of machines is used for speeding up the process, or *both solutions*, i.e., they provide both a centralized and a parallel solution. Furthermore, we can divide the tools according to the output that they produce, i.e., a *Schema Matching* approach produces always *Schema* mappings, an *Instance Matching* approach creates always *Instance* mappings (see Section 5.4.3), whereas some tools produce both instance and schema mappings. Moreover, we can distinguish the approaches according to the types of evidence (which were presented in Section 4) that they exploit for creating semantic bindings, and the resources that they use, i.e., only datasets' content (e.g., triples of datasets), or/and external resources (e.g., lexicons, translators) for improving the accuracy of mappings.

Approaches. Starting from the area of *Relational databases*, References [146, 147] have proposed a generic integration methodology (which can also be adopted by semantic web approaches) that concern the automatic resolution of conflicts (e.g., structural conflicts) and automatic generation of mappings among different schemas (or views) and the integrated schema by taking into consideration the underlying semantics of the different schemas. Fully automatic methods have also been proposed in various contexts, including query-oriented integration of relational databases [7] and methods for creating automatically mappings between taxonomy-based sources [155]. We mention eight ontology matching tools, which are categorized in Table 2 by using the criteria of Figure 8. For reasons of space, a more detailed description of each one of these is available in the `online supplementary material` (see Section A).

Evaluation Collections. The most popular benchmark (and competition) is the OAEI (Ontology Alignment Evaluation Initiative), which is a yearly evaluation event (http://oaei.ontologymatching.org/). It contains several test datasets, where each one focuses on different difficulties of ontology matching process, such as matching ontologies of the same or different domains, ontologies using different languages, and others. Finally, References [117, 144] survey over 60 *Ontology Matching* tools and a number of benchmarks for comparing the performance of such tools.

5.4.3 Instance Matching (or alignment). It refers to the problem of determining mappings at the data level, between real world entities (or instances/individuals), e.g., Thunnus Albacares ≡ Yellowfin Tuna, since they refer to the same marine species.

Context. An integration system can (a) use *predefined* (PD) instance mappings (and possibly their closure) that have been declared (e.g., axiomatically) before the integration process or/and (b) exploit *instance matching* techniques (IMT) for producing new mappings during the integration process. A materialized approach can use any of these approaches, while a virtually integration system usually relies on existing equivalence relationships and on their closure (e.g., traversal-based engines follow `owl:sameAs` paths). The most commonly used mapping (or relationship) is the `owl:sameAs` relationship, i.e., an entity e_1 is same as entity e_2 (e_1 `owl:sameAs` e_2) when both entities refer to the same real entity, e.g., http://www.dbpedia.com/Thunnus_Albacares `owl:sameAs` http://www.ecoscope.com/thunnus_albacares, while such mappings, among the URIs of marine species *Thunnus Albacares*) are shown in Figure 3 (i.e., in the *Materialized* approach). Comparing to *Schema Matching*, *Instance Matching* process produces mappings for instances only (i.e., instances and schema concepts are two disjoint sets), whose number is usually much larger than the number of schema concepts [106]. However, both processes can be very complex due to several difficulties (e.g., different conceptualizations), while Reference [23] provides more details about these two techniques.

Difficulties. It is related to the existence of many URIs for the same real entity (difficulty (b)).

Categorization. We categorize these tools by using the same criteria described in Section 5.4.2, which are shown in Figure 8. However, we can further categorize instance matching tools according to the mappings that they support (see Figure 8). Most tools usually produce `owl:sameAs` relationships, while some tools are also capable to produce other relationships, such as **foaf:basednear**, which declares that an entity has similar meaning with an other one but not exactly the same.

Approaches. Table 2 (i.e., see the last eight tools) shows scalable tools performing instance matching, which are briefly described below.

Silk [161] supports manually specified rules and supervised learning for producing `owl:sameAs` links (by default) or other user-specified relationships. It uses mainly several string similarity metrics, while it provides a parallel version for scaling out to very big datasets. Finally, *Silk* is used as a component from three materialized tools, i.e., ODCleanstore [83], *MatWare* [156], and LDIF [139].

LIMES [111] is a tool that supports both manual configuration and learning techniques (supervised and unsupervised). It offers different approximation techniques based on metric spaces for estimating the similarities between instances. It can produce both `owl:sameAs` and user-specified links, it offers both a centralized and a parallel version, while it is used by *TopFed* [133].

PARIS [148] can detect `owl:sameAs` relationships by exploiting functional properties. This system (which is used by *RapidMiner LOD Extension* [127]) does not require a configuration from the user and offers a centralized solution that has been tested with a large number of triples and entities.

WebPie [158] can take as input `owl:sameAs` relationships and computes in a parallel way their transitive and symmetric closure to produce inferred `owl:sameAs` relationships.

LINDA [15] is a fully automatic system offering both a centralized and a parallel version and has been tested for over 100 million entities. It uses several techniques for identifying `owl:sameAs` relationships, while it checks the neighborhood of different entities for inferring relationships.

MinoanER [42] is a system that discovers `owl:sameAs` relationships by placing similar descriptions into blocks, while it tries to discover mappings by using descriptions that occur in the same block. It has been built by using parallel frameworks and has been tested with millions of entities.

CoSum-P [169] is a generic framework that performs entity matching by solving a multi-type graph summarization problem. It receives an RDF graph and creates a summary graph, while it uses several structural similarity metrics, such as the number of common neighbors.

MINTE [26] (which is a component of *FuhSen* [25]) computes similarities between RDF molecules for matching equivalent entities by using semantic similarity functions, e.g., GADES [149], while except for perfoming instance matching, it integrates also all the triples of a specific real entity.

Recently, there is trend for approaches that use embeddings and Machine Learning techniques for finding similarities. *MateTee* [102] is an approach that creates a vector representation of entities and uses a Stohastic Gradient Descent method for computing similarities among entities. *RDF2VEC* [128] converts an RDF graph in sequences of entities and relations, and adapts neural language models, such as word2vec [97]. The produced vectors can be exploited for Machine Learning tasks (e.g., for identifying similarities between two entities). Finally, the task of matching different entities can become more complicated, since there exists a remarkable percentage of unnamed entities (e.g., see the example with the book in Figure 5), i.e., blank nodes. For such a case, there have been proposed techniques based on signatures [88] for blank nodes matching.

Evaluation Collections. One can exploit the *OAEI* benchmark (as in the case of *Ontology Matching*) for evaluating the performance of *Instance Matching* systems with respect to various difficulties, such as detecting differences on the values or/and the structure of different datasets that contain information about the same entities. Moreover, it contains a track, which can be used (a) for both *Ontology* and *Instance Matching*, and (b) for creating mappings not only between pairs but also among triads of datasets (http://oaei.ontologymatching.org/2018/knowledgegraph). Finally, Reference [31] contains several real and synthetic benchmarks for evaluating instance matching systems.

5.4.4 Query Answering in Virtual Integration. It refers to the process of answering of a given query in virtual integration.

Context. In Figure 3, one can see specific examples of answering a query by using a mediator, a federated query engine and a traversal-based approach.

Difficulties. For a mediator, the task of query rewriting requires the computation of mappings between the mediated schema and the underlying datasets, thereby, the heterogeneity in terms of schema, i.e., difficulty (a), and different conceptualizations, i.e., difficulty (e), can make this process more difficult. For a federated query or a traversal-based engine, it is important to select datasets that use the same model or/and URIs (they should be dereferencing in a traversal-based case), since they do not produce mappings, thereby, i.e., difficulties (a) and (b) should be overcome.

Approaches. We introduce approaches concerning the *Query Answering* process for each different virtual integration type.

Query Answering over a Mediator. In the database field, there exist two main approaches that can be defined for this process with respect to the way that the mappings have been defined. Specifically, they are called *LAV* (local as view) and *GAV* (global as view) while there exists approaches containing a combination of them, i.e., *GLAV* (global local as view) [22, 92]. *LAV* is an approach where "the source structures are defined as views over the global schema" [165] and most of the proposed solutions are based on query answering by using query rewriting, i.e., after

the query rewriting process the query can be evaluated over the underlying sources for obtaining the results. *GAV* is an approach where "each global concept is defined in terms of a view over the source schemas" [165] and most solutions try to substitute each global relation with its corresponding definition concerning the sources. *GLAV* is an approach specifying the mappings between the source schemas and the global one. Finally, Lenzerini, Calvanese, and colleagues have extensively studied the aforementioned *LAV*, *GAV*, and *GLAV* techniques and have surveyed a lot of approaches using these techniques [22, 92].

The above techniques have been also applied in the context of Linked Data [100]. In particular, *SemLAV* [101] is a LAV-based approach for processing in a scalable way SPARQL queries, by producing answers for SPARQL queries against large integration systems, whereas in Reference [89], GAV SPARQL views are exploited for rewriting queries against a global vocabulary. Regarding other query rewriting approaches, in Reference [157] techniques for query rewriting over taxonomy-based sources were proposed while in Reference [95] the authors proposed a framework performing query rewriting for SPARQL queries. It takes as input a SPARQL query represented by a specific ontology, and transforms it to a semantically equivalent query, by using an other ontology. Finally, in Reference [27] a technique is described for exploiting transformations between RDF graphs for enabling query rewriting.

Query Answering over a Federated Query Engine. The proposed engines can be distinguished in three different categories [130], according to the approach that they follow for selecting the datasets that can answer a given sub-query. First, there exists *Catalog/index-assisted* approaches, where they maintain a catalog for the available endpoints while indexes have been created and statistics have been collected during the construction of the virtually integrated system. Then, for each query the aforementioned indexes and metadata are exploited for selecting the most relevant datasets for a given triple pattern. Second, *Catalog/index-free* solutions maintain a catalog for the endpoints, however, they collect statistics on-the-fly (e.g., by exploiting SPARQL ASK queries) and the datasets are selected through the aforementioned statistics. Finally, *Hybrid solutions* combine both techniques for selected the datasets that can answer the initial query. Concerning *Catalog/index-assisted* approaches, *DaRQ* [122] system uses an index containing service descriptions for each source, which include the triple patterns that are answerable by each source. As regards *Catalog/index-free* approaches, *FedX* [140] system sends one SPARQL ASK query per triple pattern to the federated datasets' endpoints for detecting relevant datasets, e.g., for a triple pattern {?s foaf:name ?name}, it send the following query to each dataset D_i: "ASK {?s foaf:name ?name}", which returns true if D_i contains triples with this property. Finally, for decreasing the execution time of source selection task, query approximation techniques have been proposed [62].

On the contrary, most tools use a hybrid approach for answering a given query. In particular, *SPLENDID* [60] uses an index containing VoID descriptions for finding the candidate sources for answering a triple pattern, however, for triples containing variables that are not included in the VoID statistics, it sends SPARQL ASK queries. *HiBISCuS* [131] exploits an index for the subjects and the objects and SPARQL ASK queries for the predicates for discovering the relevant datasets for a given triple. *ANAPSID* [3] uses a catalog containing the available SPARQL endpoints and their ontology concepts, while statistics for the endpoints are updated on-the-fly to maintain up-to-date information. DAW [132] constructs indexes that contain information about a distinct predicate of a specific dataset (e.g., how many triples contain this predicate), and uses a novel source selection algorithm for ranking the sources based on their contribution, to select the most relevant sources for a given triple pattern. MULDER [45] sends SPARQL queries for collecting descriptions of RDF molecules templates, i.e., "descriptions of entities of the same RDF class." It mainly exploits that descriptions to select the datasets that can increase the completeness of the answer. Finally, more details about dataset selection of federated query engines and issues

that concern query optimization (e.g., query planning, join strategy) are out of the scope of this survey, however, one can find surveys containing analytical details for these steps [115, 129, 130].

Query Answering over Traversal-based Integration Approaches. *SQUIN* [69] receives a query and tries to retrieve triples by using the URIs that can be found in that query. Then, a recursive iterative process (which stops only when a fixed point is reached) that relies on the aforementioned triples starts, and in each step, *SQUIN* tries to discover incrementally more URIs and triples that are related to the given query. *Linked-Data-Fu* [66] system supports traversal-based integration, by following links based on specific rules that can be declared by using *Data-Fu* Language. It exploits REST technology, i.e., it starts by sending a POST requests to a specific URI, and the results can be derived through a GET request. By sending such requests, it can discover on-the-fly links and it can send new requests for retrieving even more data. *SWGET* uses NautiLOD formal language [53], for retrieving in a recursive way data from several linked datasets. It relies on regular expressions and ASK SPARQL queries for selecting the most appropriate sources to continue the navigation (e.g., by traversing `owl:sameAs` paths), while one can control the navigation through SPARQL queries. Moreover, we should mention *SPARQL-LD* [48], an extension of SPARQL 1.1 that enables to combine in a single query (and perform whatever transformation is possible via SPARQL) data coming not only from external SPARQL endpoints but also from external data stored in RDF, JSON-LD files, as well as data from deferenceable URIs, and others.

Query Answering over Hybrid-Integration Approaches. *TopFed* [133] is a hybrid approach (mainly a federated query engine) that contains biological data. We categorize *TopFed* as a hybrid approach, since data are transformed for being described by the same schema and they are linked with external sources through schema/instance matching techniques. For answering a specific query, this engine uses both a metadata catalog and ASK queries. *FuhSen* [25] takes as input a keyword query and a similarity threshold for creating a knowledge graph at query time. It uses a global vocabulary (called OntoFuhSen) for transforming the initial query to a SPARQL query or to a REST request, and sends federated queries to the relevant sources. Finally, the responses of the queries are enriched, by integrating all the triples of a specific entity (by using MINTE [26]). *RapidMiner LOD Extension* [127] is partially a traversal-based approach. It follows paths of a given type (e.g., `owl:sameAs` paths) for finding and integrating data over different sources; however, it can fetch data and perform instance/schema matching, while it offers data fusion mechanisms.

Evaluation Collections. There exists benchmarks like *FedBench* [137], LargeRDFBench, SP^2Bench, and others [129], which cover several dimensions such as the result completeness, ranking of the returned answers and efficiency of each different step of the process (e.g., source selection).

5.5 Auxiliary Services

This set of services are auxiliary and can be exploited before or after the integration process. In Section 5.5.1, we analyze issues concerning data *Provenance*, while in Section 5.5.2, we introduce ways to measure and improve the *Quality* of one or more datasets. In Section 5.5.3, we show how one can *Monitor* the *Evolution* of datasets, whereas in Section 5.5.4, we discuss the process of *Publishing* an integrated dataset.

5.5.1 Provenance. "It focuses on how to represent, manage and use information about the origin of the source or data to enable trust, assess authenticity and allow reproducibility" [168].

Context. Data provenance should be preserved regardless of the selected integration approach.

Difficulties. It mainly refers to difficulty (a), i.e., the initial format of the datasets can change and the contents of a dataset can be transformed (e.g., for being compatible with a global schema). In such cases, one should respect the datasets' licenses, and record the provenance of each triple.

Categorization. There are various levels of provenance support that are usually required: "(i) Conceptual level, (ii) URIs and Values level, (iii) Triple Level, and (iv) Query level" [151]. Regarding level (i), the key point is that one can transform specific triples according to a conceptual model that models provenance; i.e., it is mainly applicable in a Materialized approach. Concerning level (ii), one can adopt the "namespace mechanism for URIs," i.e., the prefix of the URI can be exploited for providing information about the origin of the data (can be supported from any approach), while for literals one can use the extension "@Source" to the end of every literal (e.g., "Thunnus" @Dbpedia). Regarding level (iii), by storing each dataset in a separate graphspace, the origin of a triple can be obtained by asking for the graphspace containing that triple (applicable only in a Materialized approach), while N-Quads format can be exploited for storing each triple's provenance. Finally, level (iv) can be supported by offering query rewriting techniques, i.e., one can exploit the graphspaces' contents for showing the contribution of each dataset to the answer of the query. It can be supported from any integration substance (mainly for Virtual Integration approaches).

Approaches. In the first provenance challenge [103] (held on 2006), several teams selected to exploit Semantic Web technologies for creating systems to represent the provenance for a "functional magnetic resonance imaging workflow" and to answer some predefined provenance queries. A provenance model for RDF data, containing the dimensions of data creation and data access, is described in Reference [67]. Specifically, the authors analyzed the major types and relationships of each dimension. Moreover, they showed ways for accessing provenance metadata, while they introduced properties form popular ontologies (such as Dublin Core and FOAF), that can be exploited for storing provenance information. The aforementioned model was used in Reference [70] along with a method for assessing the trustworthiness of the Web data. By using that method, one can assess the timeliness of the existing provenance metadata and check whether important provenance information are missing for a given dataset. A provenance ontology, which can be used for describing the provenance for both data access and data creation is presented in Reference [71]. In particular, they describe how one can include metadata provenance information to a specific dataset by using that ontology, and how to access and query such metadata, since it is important for several tasks, e.g., for evaluating the timeliness of provenance information. A generic provenance vocabulary, called PROV Model (http://www.w3.org/TR/2013/NOTE-prov-primer-20130430/), has been standardised by the W3C community. By using the PROV Model, one can describe the main entities, agents and activities being part of the production of a specific dataset, while information about the conceptual model, the constraints and applications using the PROV Model, can be found in Reference [99].

5.5.2 Quality Assessment. "It is the analysis of data to measure the quality of datasets by using relevant quality dimensions" [168].

Context. Quality is of primary importance for any integrated system. Figure 3 shows an example where a data fusion algorithm resolved a conflict, i.e., two sources agree that the max length of *Thunnus Albacares* is 240cm, while the remaining one contains another value for that fact.

Difficulties. It is predominantly related to difficulty (d) and secondarily to difficulty (f).

Categorization. There are various quality dimensions that can be measured in the context of an integrated system. Some of these dimensions can be used for assessing the quality of a single dataset, e.g., dimensions such as completeness, accuracy, data cleaning, consistency, and others [168]. For example, the completeness of a dataset can be measured for detecting and correcting possible errors and inconsistencies. Other dimensions, such as *Data Interlinking*, *Connectivity*, and *Data fusion* require information from two or more datasets (mainly evaluated by materialized approaches), while *Virtual Integration* systems mainly evaluate their *Query Performance* [115, 129,

130]. Below, we emphasize on quality dimensions requiring two or more datasets (mainly large number of datasets); however, we mention some novel approaches for the single dataset case, since the quality of an underlying dataset can affect the quality of the whole integrated content.

Data Interlinking. "It refers to the degree up to which entities that represent the same concept are linked to each other" [168]. For evaluating this quality dimension, one can check the quality of `owl:sameAs` links, the interlinking degree of a given dataset, and others. In Reference [167], the authors distinguished two categories for dataset interlinking: interlinking of external websites, i.e., for measuring the availability of links between different sources, and interlinks with other datasets, i.e., for identifying possible mappings that are inaccurate or links containing not so useful information. Reference [136] focused on crawling a large number of datasets and on providing statistics for them. The authors described data interlinking measurements such as the degree distribution of each dataset (how many datasets link to a specific dataset). LODStats [47] retrieves thousands of number of documents for providing useful statistics about how interlinked each document is, while Reference [63] introduces network measures (e.g., interlinking degree and clustering coefficient), for assessing the quality of mappings and for detecting bad quality links (e.g., `owl:sameAs` links). Finally, there exists approaches, where one can find the number of common links [105, 109], and the number of common triples, literals and schema elements [106], between two or more datasets.

Connectivity Assessment. "*Connectivity* express the degree up to which the contents of a warehouse form a connected graph that can serve, ideally in a correct and complete way, the query requirements of a semantic warehouse, while making evident how each source contributes to that degree" [104]. *Connectivity* can occur both in schema and instance level. It is useful to measure connectivity "(a) for assessing how much the aggregated content is connected, (b) for getting an overview of the warehouse, (c) for quantifying the value of the warehouse (query capabilities), since poor connectivity can result to less expressive queries, (d) for making easier its monitoring after reconstruction, and (e) for measuring the contribution of each source to the warehouse" [104].

Data Fusion, Trust, and Fact Checking. "*Data fusion* aims at resolving conflicts from different sources and find values that reflect the real world," according to Reference [41]. In Reference [83] *conflicts* are defined as the cases where two triples, belonging in different sources, contain conflicting object values for a specific subject-predicate pair. Regarding the differences with *Data Interlinking*, the latter aims at evaluating how connected two or more datasets are, i.e., whether they contain information about the same entities, while in *Data Fusion* case, such connections among datasets have already been discovered, and the goal is to find the triples containing URIs that represent the same real-world object and transform them into a single accurate representation by resolving conflicts. In the context of integrated (mainly non-structured) data and relational databases, there exists several proposed methods and approaches, depending on multiple techniques (e.g., probabilistic-based, IR models) [93]. Regarding approaches using Semantic Web notions, Google Knowledge Vault [39] stores the information in the form of RDF triples and identifies for such triple a confidence score. However, it extracts information from both RDF sources (i.e., FreeBase) and non-RDF ones.

Concerning the different strategies of data fusion [93, 96], the simplest case is to provide to the users all the conflicted objects for a given subject-predicate pair with their provenance, and let them decide whether an object is correct or not, i.e., *User-based* approaches. Another technique called *Resolution-based*, exploits a number of functions, such as majority or average for deciding which object to keep. Moreover, *Trust-based* methods take into account the degree up to which we trust the provenance of data, i.e., dataset trustworthiness can be measured as a whole, by measuring the accuracy of all the triples of a dataset, however, it is usually difficult to know dataset's trustworthiness *a priori*. Concerning semantic integration tools, *LDIF* uses *Sieve* [96] for assessing the quality of the integrated data and exploits *Resolution-based* techniques (e.g., average), *Trust-based*

Table 3. Categorizing Existing Quality Tools According to Their Characteristics

Tool / Categ.	*LinkQA* [63]	*Luzzu* [32]	*ODCleanStore* [83]	*Sieve* [96]	*SWIQA* [56]	*RDFUnit* [86]	*MatWare* [156]	*SeaStar* [135]
Quality Dimensions [168]	Completeness, Interlinking	Consistency, Conciseness, and others	Accuracy, Completeness, Consistency, Conciseness	Completeness, Consistency, Conciseness	Accuracy, Completeness, Timeliness	Accuracy, Consistency	Connectivity, Interlinking, Relevancy	Accuracy, Interlinking, Connectivity
Sources Number	Set of Mappings	One Source	Collection of Quads	One Integrated Source	One Source	One Source	Set of Sources	One or Two Sources
Output Format	HTML	RDF	RDF, HTML	N-Quads	HTML	RDF, HTML	RDF, HTML	HTML
Progr. Lang.	JAVA	JAVA	JAVA	JAVA	-	-	JAVA	JAVA
Query Lang.	-	-	-	-	SPARQL	SPARQL	SPARQL	-

techniques (e.g., prefer data from trusted sources), and configurable metrics for deciding whether a specific value is correct or a transformation is required for improving its accuracy. *ODCleanStore* [83] offers several conflict resolution rules, where some of them select only one value among the conflicting values (e.g., MAX, ANY), while it can also compute a new value based on the conflicting values. *Fuhsen* uses *MINTE* [26], which applies three fusion policies for performing data fusion of two RDF datasets, while *RapidMiner LOD extension* [127] uses some simple data fusion operators (e.g., majority) for resolving conflicts. Finally, in Reference [58] PageRank is computed for 319 RDF datasets for offering trust measurements that are also useful for evaluating dataset interlinking.

Regarding *Fact Checking*, it can be defined as the "area which focuses on computing which subset of a given set of statements can be trusted" [57, 118]. *DeFacto* [57] is a temporal approach that checks for the validity of facts. It takes RDF triples as input facts, and it exploits the web for searching for possible proofs, by supporting multiple languages. In particular, they propose supervised learning methods that require a large volume of training data while they use a combination of trustworthiness metrics and textual evidence for measuring an evidence score for a specific fact.

Crowdsourcing and Data Quality. "Crowdsourcing refers to the process of solving a problem formulated as a task by reaching out to a large network of (often previously unknown) people" [4]. There exists two different crowdsourcing mechanisms, called *content-based* and *micro-task* [4]. *Content-based* crowdsourcing concerns a group of Linked Data experts. In Reference [87], each expert used TripleCheckMate tool, which enables users to select specific DBpedia resources [90], to detect problems for the triples containing that resources, and to categorize them by using a predefined set of quality issues (described in Reference [168]). *Micro-task* mechanism concerns anonymous (even non-experts) users, thereby, the authors decided to restrict the scope of the possible errors that these users can detect [87]. For a specific triple, they can identify datatype or language errors, and incorrect links and object values (e.g., a person's birth place). Afterwards, for any mechanism that tool stores the incorrect triples (classified according to their quality issue) for further checking. Regarding other approaches, HARE [2] is a SPARQL query engine that exploits *micro-task* mechanism for completing missing values, which can occur at query execution time. Moreover, such a mechanism is used from ZenCrowd [33] for improving the quality of instance mappings (e.g., detecting incorrect `owl:sameAs` links), and from CrowdMap [134] for improving ontology alignment process.

Tools assessing the quality of RDF datasets. Table 3 lists and categorizes various tools for assessing the quality of RDF datasets. For reasons of space, a more detailed description of each one of these is available in `the online supplementary material` (see Section B).

5.5.3 Dynamics/Evolution and Monitoring. "*Dynamics* quantifies the evolution of a dataset over a specific period of time and takes into consideration the changes occurring in this period" [36].

The objective of *Monitoring* is to observe and check the progress or quality of an integrated access system over a period of time.

Context. Since everything changes very fast, any integration system should take this dimension into account. For instance, we see in Figure 3 an example where the schema of a dataset changed, thereby, the mappings in a materialized or in a mediator approach should be regenerated.

Difficulties. It is predominantly related to difficulty (f) and secondarily to difficulty (d).

Categorization. We can distinguish the integration approaches in three categories according to how they are affected when a dataset changes: (a) approaches that needs to be updated *manually*, (b) approaches that can be semi-automatically updated (e.g., by modifying a part of a configuration file and by pushing a "reconstruction" button) and (c) approaches that are *automatically* updated (i.e., not affected from datasets' updates). Moreover, datasets' evolution can occur in *ontology level* and *instance level*. Evolution in ontology level occurs when the schema of at least one dataset changes or the global schema changes. By using a global schema, only the mappings between that dataset and the global schema should be recreated (i.e., in a mediator or/and in a warehouse). However, the construction of pair-wise mappings results to the recreation of the mappings between the aforementioned dataset and every other dataset (i.e., in a warehouse). For *Catalog/index-assisted* or *hybrid* federated query approaches (such as DaRQ [122]), the indexes containing statistics for properties/classes should be refreshed, while approaches such as FedX [140] (which relies on ASK queries), or ANAPSID [3] (which collects statistics on-the-fly) are automatically updated. On the contrary, evolution in instance level occurs when the policy (e.g., the prefix) of a dataset's URIs changes, or when more URIs are added in a specific dataset. For a materialized approach, instance matching rules should be reconstructed and all the mappings containing the dataset that changed should be recreated. Regarding virtual integration, it is essential to update indexes or/and statistics, that are affected through such an evolution (e.g., the number of triples where a property occurs can be changed when more URIs are added). Finally, by storing in a warehouse the integrated content as a single integrated dataset, the evolution of a dataset can result to its whole reconstruction. For avoiding this time consuming process, one can store the datasets in different graphspaces (e.g., like in MarineTLO warehouse [151]) and update only the dataset(s) that changed.

Approaches. In Reference [85], the authors provide an approach that allows query answering in virtual integration systems under evolving ontologies without recreating mappings between the mediator and the underlying sources. In particular, it can be achieved by rewriting queries between different ontology versions and then forwarding them to the underlying sources to be answered. Moreover, in Reference [79] a "Dynamic Linked Data Observatory" is introduced for monitoring Linked Data over a specific period of time, whereas Reference [36] proposes a function for measuring the dynamics of a specific dataset, while they list several approaches related to datasets' dynamics. *SPARQLES* [159] and *SpEnD* [166] monitor the health of hundreds of public SPARQL endpoints, by sending SPARQL queries at regular intervals, e.g., for checking the availability of each endpoint over time. Finally, incosistencies can occur in the *specificity* of ontological instance descriptions, when such descriptions are migrated to the up-to-date version of a given ontology [152].

5.5.4 Publishing an Integrated Dataset. Its objective is to publish an integrated dataset to be reused by others for several tasks (e.g., query evaluation, data analysis, etc.).

Context. A scientist/user can publish an integrated dataset to be findable and reusable by others. However, Linked Data are not so "open" for the consumers or the scientists to be reused, since in many cases they are published under a license. According to References [54, 73], each RDF dataset should contain such a license in order the datasets to be reused under legal terms. Moreover, the license should be both machine-readable (e.g., mentioned in the VoID description of a dataset) and human-readable (e.g., mentioned in the documentation of the dataset). According

to Reference [54] a license should contain information for the permission that concern the reproduction, distribution, modification or redistribution. Therefore, in case of publishing an integrated dataset, the publishers should respect both the provenance and the licenses of the constituent datasets. Such information should be included in the metadata of the dataset and published along with the dataset. Moreover, the publisher of any dataset (e.g., integrated dataset) should use recommended standards and follow specific principles, such as Linked Data principles [12] and FAIR principles [164], for making their dataset more discoverable, reusable and readable from both "'humans and computers." In Figure 3, we can see some ways for publishing an integrated dataset in the warehouse approach.

Difficulties. Datasets are produced by several organizations in different licenses, places, schemas, formats, and so forth (difficulty (a)).

Recommended Standards for Data Publishing. There exists several W3C standards that are recommended to be used during the creation and before publishing any dataset (a single or an integrated one), for enhancing *Data Exchange* and *Data Integration*. The key standards can be divided in (a) standards for creating links among a single dataset and other datasets (e.g., relationships between instances and ontologies), and (b) standards for creating basic metadata for a dataset (e.g., description, name, language, provenance). Concerning (a), it is important each dataset to contain links to (schema/instance) URIs of other datasets, by using standard vocabularies such as *OWL*, *RDF/RDFs*, *SKOS* (http://www.w3.org/2004/02/skos/) and so forth. Regarding (b), the basic metadata of each dataset should be desribed by using standard vocabularies, such as *DCAT* (http://www.w3.org/TR/vocab-dcat/), *schema.org*, *VoID* (http://www.w3.org/TR/void/), *Dublin Core* (http://dublincore.org), *FOAF* (http://xmlns.com/foaf/spec/), *PROV* [99], and so on. Indeed, there is an emerging trend of using such vocabularies in several cases, i.e., they are used by billions of *Web Pages* [18, 121] (mainly through Microdata or/and JSON-LD), by *Digital Libraries* [145], and by *Governments* [35], e.g., Open Government Data from European Union, Unites States, United Kingdom, and others, usually through the creation of descriptions using schema.org or/and DCAT vocabulary. Furthermore, Google's dataset search engine collects and indexes datasets' metadata that have been expressed by using such ontologies [18]. Therefore, it is clearly of primary importance for any publisher to use these standards, for enabling data sharing, and for easing the integration of their dataset with other ones. Moreover, it is worth noting that *VoID* vocabulary can be exploited for deciding whether two or more datasets are worth to be integrated, since it can be used for describing metadata concerning the links that exist between two datasets. Finally, we should also mention the Shapes Constraint Language, i.e., *SHACL* (http://www.w3.org/TR/shacl/), which is a language for "validating RDF graphs against a set of conditions," that can used for several purposes, such as for data cleaning before publishing the dataset.

Where to Publish Datasets. First, dataset catalogs such as *datahub* (http://datahub.io/) and *Zenodo* (http://zenodo.org/), offer services for uploading a dataset, metadata about the dataset and so forth. Second, a SPARQL endpoint can be exploited in order the dataset to be directly published and accessed by humans and machines (e.g., for query answering). In particular, there are several available tools offering query evaluation such as *Virtuoso* (http://virtuoso.openlinksw.com/), *BlazeGraph* (http://www.blazegraph.com), *AllegroGraph* (http://www.franz.com/agraph/), and *Stardog* (http://www.stardog.com/), which has been successfully used in *NASA MARS mission* (see more details in https://cdn2.hubspot.net/hubfs/2820685/Assets/Case Studies/Stardog_Nasa Case Study.pdf).

5.6 Integration Tools

Table 4 lists a number of integration tools and categorizes them according to the dimensions of Section 4, i.e., the integrated method that they offer, their input and output types and what internal

Table 4. Categorizing Existing RDF Integration Tools

Tool/ Framework	Integration Substance	Dataset Types	Output Types	Transformations	Schema Matching	Instance Matching	V Q A	Provenance Levels	Quality	Evolution	Tested |D|	Tested |T|
LDIF [139]	Materialized	RDF	Any	LT	PD+OMT	PD+IMT	✗	CL,UVL,TL	DF	S-Aut.	1–9	B
ODCleanstore [83]	Materialized	RDF	Any	LT	PD+OMT	PD+IMT	✗	CL,UVL,TL	DF	S-Aut.	1–9	M
MatWare [156]	Materialized	RDF+O	Any	LT, FT	PD+OMT	PD+IMT	✗	CL,UVL,TL	Con.	S-Aut.	1–9	M
KARMA [84]	Materialized	RDF+O	Any	LT, FT	PD+OMT	PD	✗	CL,UVL,TL	DC	S-Aut.	1–9	B
FuhSen [25]	Hybrid	RDF+O	KS	FT	PD	PD+IMT	✓	UVL,QL	DF	S-Aut.	1–9	M
TopFed [133]	Hybrid	RDF	QA	LT, FT	PD+OMT	PD+IMT	✓	UVL,QL	QP	S-Aut.	10–19	B
RapidMinerLOD [127]	Hybrid	RDF+O	Any	LT, FT	PD+OMT	PD+IMT	✓	UVL,QL	DF	Aut.	1–9*	M
SQUIN [69]	Traversal	RDF	QA	✗	PD	PD+C	✓	UVL,QL	QP	Aut.	1–9*	M
SWGET [53]	Traversal	RDF	QA	✗	PD	PD+C	✓	UVL,QL	QP	Aut.	1–9*	M
Linked-Data-Fu [66]	Traversal	RDF+O	Any	✗	PD	PD+C	✓	UVL,QL	QP	Aut.	10–19*	M
SEMLAV [101]	Mediator	RDF	QA	✗	PD+OMT	PD	✓	UVL,QL	QP	S-Aut.	1–9	M
DaRQ [122]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	S-Aut.	10–19	M
Splendid [60]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	S-Aut.	10–19	B
HiBISCuS [131]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	S-Aut.	10–19	B
FedX [140]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	Aut.	10–19	B
ANAPSID [3]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	Aut.	10–19	B
DAW [132]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	S-Aut.	1–9	M
MULDER [45]	Federated	RDF	QA	✗	PD	PD	✓	UVL,QL	QP	S-Aut.	10–19	M

O = Other Formats, Any = Any Service, QA = Query Answering, KS = Keyword Search, FT = Format Transformation, LT = Logical Transformation, PD = Predefined Mappings, OMT = Ontology Matching Techniques, C = Closure, IMT = Instance Matching Techniques, VQA = Virtual Query Answering, CL = Conceptual Level, UVL = URIs and Values Level, TL = Triples Level, QL = Query Level, DF = Data Fusion, Con. = Connectivity, DC = Data Cleaning, QP = Query Performance, Aut. = Automatically, S-Aut. = Semi-Automatically, |D| = Datasets, *discovers more datasets on-the-fly, |T| = Triples, M = Millions, B = Billions.

(e.g., instance matching) and auxiliary services (e.g., provenance) they support. Although these tools can support millions or even billions of triples, to the best of our knowledge, they have been tested by using a small number (below 20) of real or synthetic datasets (see the last two columns of Table 4). Materialized approaches have mainly been tested by using real datasets [84, 139, 156], while federated approaches have been evaluated by using benchmarks such as LargeRDFBench [129] (having billions of triples) and FedBench [137] (having millions of triples). Materialized approaches are difficult to scale up to a big number of datasets, since some steps require manual effort, e.g., defining and configuring matching/transformation rules. On the contrary, virtual integrated systems do not offer transformation and data fusion mechanisms and mainly rely on a common schema (or/and common URIs), therefore, conceptualization, naming and conflicts issues are difficult to be tackled. Finally, when a dataset evolves, some steps that possibly require manual effort (e.g., defining new matching rules) should be repeated for most integration tools.

6 PROCESSES FOR INTEGRATION

In the previous sections, we have factorized the integration problem to various dimensions, and we have analyzed the approaches and methods for each one of these dimensions. Regarding *InternalServices* dimension, one question is what processes (i.e., sequence of steps) are usually followed. For instance, given two or more sets of triples to be integrated, does one start from ontology matching or from instance matching? Of course, various processes could be followed according to the context. Below, we distinguish the main ones, each accompanied by a real-world case.

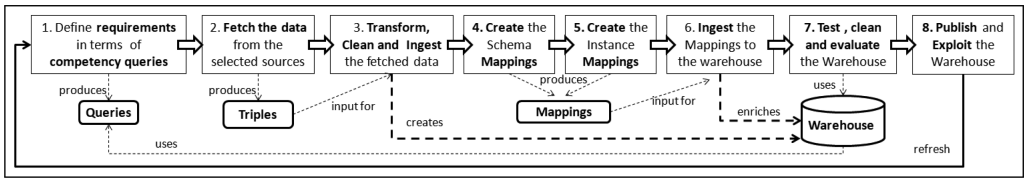

Fig. 9. Top level—ontology-based integration.

P1. (Processes for) Top-level Ontology-based or Competency Query-based Integration: Here, the desired Level III requirements are specified either as competency queries and/or by providing the ontology or schema that the integrated view of the datasets should have. Then, the data of the individual datasets should be transformed (physically or virtually) according to the integrated schema. Indeed, information integration is traditionally done in the context of databases by reconciling data coming from different sources under a common schema. An analogous process for the case of the RDF data (that follows the materialized approach), is the one that is followed by *LDIF* [139] and *ODCleanStore* [83], which are generic frameworks for integrating Linked Data. *MatWare* [156] follows a similar process and it has been used for building and maintaining real and operational warehouses, i.e., MarineTLO warehouse [151] and the ongoing GRSF warehouse [154]. In Figure 9, one can observe the process that is usually followed by such frameworks and tools. In general, we could say that the process starts with the specification of a kind of "integration template" that the data of the underlying datasets should "fill." This template can be specified by (competency) queries, schemas/ontologies or both.

P2. (Processes for) General purpose integration (fully automatic): In this case, we do not have an "integration template," either because we are not aware about the contents of the underlying datasets, and/or because the integration does not have a specific purpose, i.e., we aim at building a general purpose integrated view of a set of datasets. Therefore, here we try to do the best that we can, and usually in such cases it is more difficult to guarantee that the integrated view satisfies the criteria of completeness, validity and accuracy. We can say that approaches and systems that fall into this category include *LODLaundromat* [126], *LOD-a-Lot* [51], and *LODsyndesis* [105, 107]. In these systems the process followed varies.

Specifically, *LODLaundromat* [126] starts by collecting URLs denoting dataset dumps and downloads the datasets by connecting to the hosting servers. Afterwards, it performs data cleaning by finding and correcting syntax errors, by replacing blank nodes with well-known URIs and by removing duplicate triples. Then, it stores the documents in a uniform serialization format (each document is stored in a different file), it creates the metadata for the underlying datasets and it publishes the data for being reused for various purposes in N-Triples and HDT format [52]. *LOD-a-Lot* [51] is a service that integrate all the documents of *LODLaundromat* (over 650K documents) into a single indexed HDT file [52] for easing the process of accessing and querying the full corpus as a whole. *LODsyndesis* [105, 107] starts by collecting hundreds of available datasets from online catalogs (such as datahub.io) and then it computes the transitive and symmetric closure of `owl:sameAs` relationships to find all the equivalent entities among the datasets. Afterwards, it creates indexes and performs measurements that are exploited for offering several services such as object coreference dataset discovery and selection, connectivity assessment and others. Figure 10 illustrates an indicative lifecycle model of an integrated dataset in the form of UML State Diagram. Each integration process can be conceived as a series of transitions between the illustrated states.

P3. Composite Processes: Composite processes are also possible. A composite process can be seen as a sequence of points of the multidimensional space that we defined. For example, suppose an

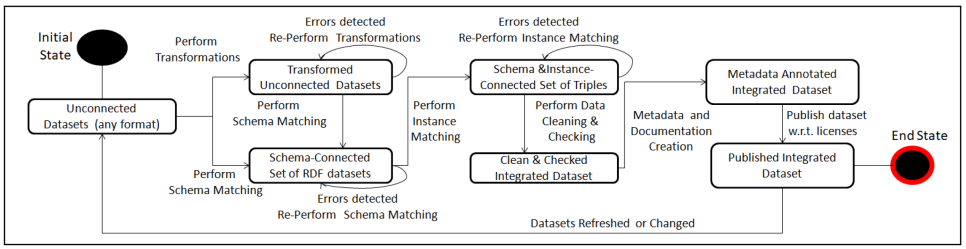

Fig. 10. Lifecycle of an integrated dataset.

integration process that comprises two subprocesses: the first aims at discovering and selecting the 10 most related datasets to one information need, and the second aims at five-level integration of these 10 datasets of the first step. In general, workflow management systems could be exploited for specifying and enacting complex processes for data integration.

Remark about evaluation. It is easier to evaluate how successful a P1 process was. It is harder for P2 processes and even harder for P3 processes.

7 EVALUATION OF INTEGRATION

How an integration approach can be evaluated? In general, we could identify two basic evaluation aspects, *quality* and *cost* (the latter includes the human effort required), and ideally we would like optimal quality and fully automatic (and cost-effective) integration processes. Of course, an integration process comprises various steps, each one could be evaluated independently of the others, and in Section 5 we referred to such benchmarks. However, there is a need for evaluating also the *overall process* and its outcome, and for this reason below we discuss how each kind of services (that can be delivered through an integration process), either fine-grained or coarse-grained (as distinguished in Section 5.2), can be evaluated.

Evaluation of Fine-grained Services. Here, we describe ways and we introduce evaluation collections concerning the fine-grained services.

- **FG: Level I.** For offering complete *Global URI Lookup Services*, i.e., for finding all the datasets and the equivalent URIs of a URI u, it is important to produce high quality mappings among URIs and to offer cross-dataset completion (see Section 5.2.1). Incorrect `owl:sameAs` mappings result to low quality services, i.e., URIs referring to different entities are considered as equivalent. The quality of mappings is mainly evaluated by using metrics such as precision and recall [63], whereas cross-dataset completion is evaluated through connectivity metrics [104, 105]. Concerning the cost, several tools automate such processes and produce mappings having high precision (see Sections 5.4.2 and 5.4.3); however, they are not free of errors. Consequently, human effort is required for interpreting the results of the measurements and the produced mappings. For this reason, crowdsourcing approaches are used for improving the precision and recall of matching process [2, 4, 33, 87, 134].
- **FG: Level II.** The evaluation of *Global Keyword Search Services* relates more to *Information Retrieval techniques*. In particular, a straightforward way for evaluating such a service is to use reference collections [138]. LUBM benchmark [64] contains an evaluation collection with keyword queries (and their corresponding answer) that can be evaluated over RDF datasets. *Concerning Virtual Integration Systems*, there exists benchmarks like *FedBench* [137], *LargeRDFBench*, SP2Bench, and others [129], which cover several dimensions, such as the result completeness, ranking of the returned answers, and efficiency (e.g., execution time). INEX Linked Data track provides a collection, which contains Wikipedia articles,

and *DBpedia* and *YAGO* links (and triples) for the entities that occur in each article [43]. The main task of that track was to find the most relevant articles for a keyword query. Finally, in *SemSearch* workshops the challenge was to evaluate semantic web *Keyword Search* engines at large scale, by using the "Billion Triples Challenge 2009" dataset, which contains 1.4 billion of triples [43].

- **FG: Level III.** The evaluation of *Integrated Query Answering Services* differs from the previous level, since the input for such an evaluation is a query expressed in a structured query language. One way to evaluate such an approach, is to predefine some competency queries [104] (MarineTLO warehouse [151] and GSRF warehouse [154] are evaluated in this way). "A competency query is a query that is useful for the community at hand, e.g., for a human member (e.g., a scientist)" [104], and sketches the desired scope and structuring of the information before the creation of the integrated system. For evaluating correctly each competency query, the expected results should be predefined. However, this type of evaluation requires human effort for defining the requirements of the integrated access system. Concerning *Virtual Integration Systems*, in Section 5.4.4 we mentioned several benchmarks for evaluating such systems. General question answering benchmarks are also used for testing such services, e.g., *Free917* [19] is an open question answering benchmark containing 917 natural language questions that can be answered by using the content of Freebase dataset. Moreover, *QALD* [94] is an evaluation series campaigns on question answering over Linked Data. The major challenge for the participants, is to use as input several RDF datasets and natural language questions, for returning the desired answers or a SPARQL query that is able to produce the correct answers. *QALD* is used for evaluating several aspects such as *multilinguality*, *large-scale question answering* and others, while for each aspect it contains several questions and tasks. Finally, *LC-QuAD* [150] is a corpus containing 5,000 questions for testing several different cases, while it provides the corresponding SPARQL queries that are essential for answering questions over DBpedia [90]. Finally, a framework is provided for generating natural language questions and their corresponding SPARQL queries for reducing the manual effort.

Evaluation of Coarse-grained Services. Concerning *Dataset Discovery and Selection Services*, the results of such services for a specific integration task, are usually evaluated by measuring the marginal gain of the selected sources [40], or/and by measuring the level of connectivity among any set of datasets [106], i.e., for evaluating whether the selected datasets contain information for the same entities. Additional quality dimensions can be considered, such as timeliness, accuracy, coverage and efficiency [125], as well as constraints such as the cost of integration (e.g., in terms of money amount for private datasets).

8 SEMANTIC INTEGRATION ON A LARGE SCALE

Table 5 compares the running LOD cloud tools/services that contain information on over 100 LOD datasets, according to the services that they provide (each service can be exploited for different dimensions of Data Integration), while each tool/service is described in more detail in `the online supplementary material` (see Section C).

Below, we mention some real, and noteworthy, cases of integration in a large scale, and we divide them in *domain-specific* and *domain-independent* cases. These cases either contain large number of datasets, or the number of datasets is not big but the integrated dataset is itself big (and popular).

Domain-Specific Integration Use Cases. Here, we introduce use cases of successful data integration that have been applied for a specific domain. *Europeana* [76] is a digital library for cultural heritage combines data from than 3,000 institutions across Europe while these data were

Table 5. Existing Services for Large Numbers of RDF Datasets

Tool/Service	Total Triples	Include > Datasets	Global URI Lookup	Dataset Discovery	Dataset Visual-ization	Connectivity	Fetching Trans-forming	Keyword Search	Dataset Analysis	Querying Datasets	Dataset Evolution
LODLaundromat [126]	38 Bil.	>100,000*	✓	✓			✓	✓			
LOD-a-lot [51]	28 Bil.	>100,000*					✓		✓	✓	
LODStats [47]	130 Bil.	>1,000		✓					✓		
LODsyndesis [105]	2 Bil.	>100	✓	✓	✓	✓					
Datahub.io	Unk.	>1,000		✓			✓		✓		
LinkLion [109]	77 Mil.	>100		✓		✓	✓				
DyLDO [79]	Unk.	>10,000*									✓
LODCache	4 Bil.	>10,000*						✓		✓	
LODCloud [136]	Unk.	>1,000			✓	✓	✓		✓		
sameAs.org [59]	Unk.	>100	✓								
SPARQLES [159]	Unk.	>100		✓	✓						✓
SpEnD [166]	Unk.	>100		✓	✓						✓

*Documents, Mil. = Million, Bil. = Billion, Unk. = Unknown.

transformed into Linked Data (over 1.8 billion of triples). One can query the integrated content by using SPARQL queries. *OCLC* (http://www.oclc.org) is a "global library cooperative" that supports data from thousands of libraries. *OCLC* has developed *WorldCat* (http://worldcat.org), which contains 2.7 billion records with bibliographic metadata in Linked Data format. For making the data more findable by search engines (e.g., Google Dataset Search [18]) and reusable from other users, data are expressed by using standard formats and vocabularies, such as *RDFa* and *schema.org*. *PhLeGrA* [80] has integrated data from four heterogeneous large scale biomedical datasets. The authors analyzed the integrated graph for discovering associations between drugs, and they used that graph and machine learning techniques for improving the accuracy of predictions of adverse drug reactions. *Bio2RDF* is "the largest network of Linked Data for Life Sciences" [20]. It contains approximately 11 billion triples from 35 datasets, and it can execute federated queries among the underlying sources, since it creates mappings between the ontology of each dataset and a global schema, called *Semanticscience Integrated Ontology (SIO)*. Finally, *Open Phacts* [61] has integrated over 3 billion triples from approximately 20 datasets containing information about drugs, and it exploits a mediator mechanism for supporting complex SPARQL queries.

Domain Independent Integration Use Cases. Here, we introduce domain-independent use cases of successful data integration. *Google Knowledge Graph* contains over 18 billion facts for over 570 million entities [39] and exploits Semantic Web technologies for integrating information for any domain. It integrates both semi-structured data (e.g., from organizations and companies) and data from millions of HTML web pages that are mainly expressed in standard formats and vocabularies, such as *JSON-LD*, *Microdata*, and *schema.org*. Except for *Google Knowledge Graph*, *Google* uses similar mechanisms for providing integration for specific domains, such as for *Shopping* (http://developers.google.com/search/docs/data-types/product), for providing *Maps* (http://developers.google.com/search/docs/data-types/local-business) and others. *Wikidata* [162] is a free and open knowledge base that contains over 26 million entities and it is readable and editable by both humans and machines. It contains data in over 350 different languages, while it mainly integrates data from Wikipedia (different languages) and users' data. *Freebase* [16] is a former knowledge base that integrated data from four sources: Wikipedia, MusicBrainz (http://musicbrainz.org), FMD (http://www.fashionmodeldirectory.com) and the dataset of NNDB

(http://www.nndb.com), and contained approximately 2 billion triples. The users had the opportunity to contribute to the knowledge base by editing structured data. *YAGO* [124] is a multilingual knowledge base that integrates data from Wikipedia, Wordnet (http://wordnet-rdf.princeton.edu/), and Geonames (http://www.geonames.org). It includes over 1 billion triples for approximately 17 million entities. It is constructed through information extraction and merging and not by community effort. *DBpedia* [90] is a knowledge base that integrates multilingual data extracted from Wikipedia (http://www.wikipedia.org) in 125 languages. It contains over 3 billion triples, and describes over 38 million things. For easing the integration process with other sources, it contains large number of links to other datasets in the LOD cloud. In a comparative survey [50], the authors defined 35 aspects according to which knowledge bases can be analyzed, while in Reference [49] they analyzed the quality of the aforementioned knowledge bases in several dimensions.

9 DISCUSSION AND PROMISING RESEARCH DIRECTIONS

The presented integration tools (in Section 5.6) have been applied over a small number of datasets and cannot easily scale up to large number of datasets. For this reason there is a recent trend for services for several RDF datasets (as mentioned in Section 8); however, each one mainly focuses on a single aspect of the integration problem. Although there is a number of successful integration examples including domain specific (e.g., References [20, 61, 76]) and domain independent (e.g., References [39, 90, 162]), we can identify several topics that are worth further research, including: (i) *Advanced Dataset Discovery and Selection*, (ii) *Data Quality and Veracity*, (iii) *Measurements and Services in Global Scale*, and (iv) *Evaluation Collections and Reproducible Results*. In `the online supplementary material` (see Section D), we present a more detailed discussion of these topics.

10 CONCLUDING REMARKS

In this article, we described the Linked Data Ecosystem by identifying the main actors and use cases, and then we discussed the main difficulties of the data integration process. Since the integration landscape is wide and complex, we factorized the process according to five dimensions, we discussed the spectrum of binding types that are used for achieving integration, as well as the kinds of evidence that are usually used for creating such bindings. Subsequently, we used these dimensions for describing the work that has been done in the area. Apart from discussing the main works, we gave emphasis on approaches for large scale semantic integration. We identified and discussed 18 tools for linked data integration and 12 services that are available for several RDF datasets. By analyzing the integration problem and the existing methodologies and tools, we identified various directions that are worth further research, including services for large scale dataset discovery, and quality/veracity testing. Finally, we should stress that evaluation methodologies and collections appropriate for obtaining reproducible and comparative evaluation results, for the overall integration process, would be very useful for the community, since they would aid the evaluation of novel integration processes.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their comments and suggestions, which helped us to improve this article.

REFERENCES

[1] A. Abello, O. Romero, T. B. Pedersen, R. Berlanga, V. Nebot, M. J. Aramburu, and A. Simitsis. 2015. Using semantic web technologies for exploratory OLAP: A survey. *IEEE Trans. Knowl. Data Eng.* 27, 2 (2015), 571–588.

[2] M. Acosta, E. Simperl, F. Flöck, and M. Vidal. 2017. Enhancing answer completeness of SPARQL queries via crowdsourcing. *Web Semantics: Sci. Serv. Agents World Wide Web* 45 (2017), 41–62.

[3] Maribel Acosta, Maria-Esther Vidal, Tomas Lampo, Julio Castillo, and Edna Ruckhaus. 2011. ANAPSID: An adaptive query processing engine for SPARQL endpoints. In *Proceedings of the International Semantic Web Conference (ISWC'11)*. Springer, 18–34.

[4] Maribel Acosta, Amrapali Zaveri, Elena Simperl, Dimitris Kontokostas, Fabian Flöck, and Jens Lehmann. 2018. Detecting linked data quality issues via crowdsourcing: A dbpedia study. *Semantic Web* 9, 3 (2018), 303–335.

[5] Grigoris Antoniou and Frank Van Harmelen. 2004. *A Semantic Web Primer*. MIT Press.

[6] Ciro Baron Neto, Kay Müller, Martin Brümmer, Dimitris Kontokostas, and Sebastian Hellmann. 2016. LODVader: An interface to LOD visualization, analytics and discovERy in real-time. In *Proceedings of the Conference on the World Wide Web (WWW'16)*. 163–166.

[7] Sonia Bergamaschi, Silvana Castano, and Maurizio Vincini. 1999. Semantic integration of semistructured and structured data sources. *ACM SIGMOD Rec.* 28, 1 (1999), 54–59.

[8] Tim Berners-Lee and Mark Fischetti. 2001. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web by Its Inventor*. DIANE Publishing Company.

[9] Tim Berners-Lee, James Hendler, Ora Lassila et al. 2001. The semantic web. *Sci. Amer.* 284, 5 (2001), 28–37.

[10] Nikos Bikakis and Timos K. Sellis. 2016. Exploration and visualization in the web of big linked data: A survey of the state of the art. In *Proceedings of the International Joint Conference on Extending Database Technology and International Conference on Database Theory (EDBT/ICDT'16)*, Vol. 1558.

[11] N. Bikakis, C. Tsinaraki, N. Gioldasis, I. Stavrakantonakis, and S. Christodoulakis. 2013. The XML and semantic web worlds: Technologies, interoperability and integration: a survey of the state of the art. In *Semantic Hyper/Multimedia Adaptation*. Springer, 319–360.

[12] Christian Bizer, Tom Heath, and Tim Berners-Lee. 2009. Linked data-the story so far. *International Journal on Semantic Web and Information Systems*, Tom Heath, Martin Hepp, and Christian Bizer (Eds.), 5, 3 (2009), 1–22.

[13] Christian Bizer and Andreas Schultz. 2010. The R2R framework: Publishing and discovering mappings on the web. In *Proceedings of the 1st International Conference on Consuming Linked Data*. 97–108.

[14] Christian Bizer, Andreas Schultz, David Ruiz, and Carlos R. Rivero. 2012. Benchmarking the performance of linked data translation systems. In *Proceedings of the Conference on Linked Data on the Web (LDOW'12)*.

[15] Christoph Böhm, Gerard de Melo, Felix Naumann, and Gerhard Weikum. 2012. LINDA: Distributed web-of-data-scale entity matching. In *Proceedings of the 21st ACM ACM International Conference on Information and Knowledge Management Conference (CIKM'12)*. 2104–2108.

[16] Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD Conference*. 1247–1250.

[17] Angela Bonifati, Fabiano Cattaneo, Stefano Ceri, Alfonso Fuggetta, Stefano Paraboschi et al. 2001. Designing data marts for data warehouses. *ACM Trans. Softw. Eng. Methodol.* 10, 4 (2001), 452–483.

[18] Dan Brickley, Matthew Burgess, and Natasha Noy. 2019. Google dataset search: Building a search engine for datasets in an open web ecosystem. In *Proceedings of the World Wide Web Conference*. ACM, 1365–1375.

[19] Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the Association for Computational Linguistics (ACL'13)*. 423–433.

[20] Alison Callahan, José Cruz-Toledo, Peter Ansell, and Michel Dumontier. 2013. Bio2RDF release 2: Improved coverage, interoperability and provenance of life science linked data. In *Proceedings of the Extended Semantic Web Conference*. Springer, 200–212.

[21] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, Daniele Nardi, and Riccardo Rosati. 1998. Information integration: Conceptual modeling and reasoning support. In *Proceedings of 3rd International Conference on Cooperative Information Systems (IFCIS'98)*. 280–289.

[22] Diego Calvanese, Domenico Lembo, and Maurizio Lenzerini. 2001. Survey on methods for query rewriting and query answering using views. *Integrazione, Warehousing e Mining Di Sorgenti Eterogenee* 25 (2001).

[23] Silvana Castano, Alfio Ferrara, Stefano Montanelli, and Gaia Varese. 2011. Ontology and instance matching. In *Knowledge-driven Multimedia Information Extraction and Ontology Evolution*. Springer, 167–195.

[24] M. Cheatham and P. Hitzler. 2013. String similarity metrics for ontology alignment. In *Proceedings of the International Semantic Web Conference (ISWC'13)*. Springer, 294–309.

[25] Diego Collarana, Mikhail Galkin, Ignacio Traverso-Ribón, Christoph Lange, Maria-Esther Vidal, and Sören Auer. 2017. Semantic data integration for knowledge graph construction at query time. In *Proceedings of the IEEE International Conference on Semantic Computing (ICSC'17)*. IEEE, 109–116.

[26] Diego Collarana, Mikhail Galkin, Ignacio Traverso-Ribón, Maria-Esther Vidal, Christoph Lange, and Sören Auer. 2017. MINTE: Semantically integrating RDF graphs. In *Proceedings of International Conference on Web Intelligence, Mining and Semantics (WIMS'17)*. ACM, 22.

[27] Gianluca Correndo, Manuel Salvadores, Ian Millard, Hugh Glaser, and Nigel Shadbolt. 2010. SPARQL query rewriting for implementing data integration over linked data. In *Proceedings of the International Joint Conference on Extending Database Technology and International Conference on Database Theory (EDBT/ICDT'10)*. ACM, 1–11.

[28] Isabel F. Cruz, Flavio Palandri Antonelli, and Cosmin Stroe. 2009. AgreementMaker: Efficient matching for large real-world schemas and ontologies. *Proc. VLDB Endow.* 2, 2 (2009), 1586–1589.

[29] A. Dadzie and M. Rowe. 2011. Approaches to visualising linked data: A survey. *Semantic Web* 2, 2 (2011), 89–124.

[30] Mathieu d'Aquin and Enrico Motta. 2011. Watson, more than a semantic web search engine. *Semantic Web* 2, 1 (2011), 55–63.

[31] Evangelia Daskalaki, Giorgos Flouris, Irini Fundulaki, and Tzanina Saveta. 2016. Instance matching benchmarks in the era of Linked Data. *Web Semantics: Sci. Serv. Agents World Wide Web* 39 (2016), 1–14.

[32] Jeremy Debattista, Sören Auer, and Christoph Lange. 2016. Luzzu—A methodology and framework for linked data quality assessment. *J. Data Info. Qual.* 8, 1 (2016), 4.

[33] Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudré-Mauroux. 2012. ZenCrowd: Leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In *Proceedings of the Conference on the World Wide Web (WWW'12)*. ACM, 469–478.

[34] Li Ding, Tim Finin, Anupam Joshi, Rong Pan, R. Scott Cost, Yun Peng, Pavan Reddivari, Vishal Doshi, and Joel Sachs. 2004. Swoogle: A search and metadata engine for the semantic web. In *Proceedings of the Conference on Information and Knowledge Management (CIKM'04)*. ACM, 652–659.

[35] Li Ding, Vassilios Peristeras, and Michael Hausenblas. 2012. Linked open government data [Guest editors' introduction]. *IEEE Intell. Syst.* 27, 3 (2012), 11–15.

[36] Renata Queiroz Dividino, Thomas Gottron, Ansgar Scherp, and Gerd Gröner. 2014. From changes to dynamics: Dynamics analysis of linked open data sources. In *Proceedings of the European Semantic Web Conference (ESWC'14)*. Retrieved from CEUR-WS.org.

[37] Warith Eddine Djeddi and Mohamed Tarek Khadir. 2014. A novel approach using context-based measure for matching large-scale ontologies. In *Proceedings of the International Conference on Big Data Analytics and Knowledge Discovery (DaWaK'14)*. Springer, 320–331.

[38] Martin Doerr. 2003. The CIDOC conceptual reference module: An ontological approach to semantic interoperability of metadata. *AI Mag* 24, 3 (2003), 75.

[39] X. Dong, E. Gabrilovich, G. Heitz, W. Horn, Ni Lao, K. Murphy, T. Strohmann, S. Sun, and W. Zhang. 2014. Knowledge vault: A web-scale approach to probabilistic knowledge fusion. In *Proceedings of the ACM Special Interest Group on Knowledge Discovery and Data Mining (SIGKDD'14)*. ACM, 601–610.

[40] Xin Luna Dong, Barna Saha, and Divesh Srivastava. 2012. Less is more: Selecting sources wisely for integration. In *Proceedings of the VLDB Endowment*, Vol. 6. VLDB Endowment, 37–48.

[41] X. L. Dong and D. Srivastava. 2015. Big data integration. *Synth. Lect. Data Manage.* 7, 1 (2015), 1–198.

[42] Vasilis Efthymiou, Kostas Stefanidis, and Vassilis Christophides. 2016. Minoan ER: Progressive entity resolution in the web of data. In *Proceedings of the 19th International Conference on Extending Database Technology*.

[43] Khadija M. Elbedweihy, Stuart N. Wrigley, Paul Clough, and Fabio Ciravegna. 2015. An overview of semantic search evaluation initiatives. *Web Semantics: Sci. Serv. Agents World Wide Web* 30 (2015), 82–105.

[44] Mohamed Ben Ellefi, Zohra Bellahsene, Stefan Dietze, and Konstantin Todorov. 2016. Dataset recommendation for data linking: An intensional approach. In *Proceedings of the International Semantic Web Conference*. Springer, 36–51.

[45] Kemele M. Endris, Mikhail Galkin, Ioanna Lytra, Mohamed Nadjib Mami, Maria-Esther Vidal, and Sören Auer. 2017. MULDER: Querying the linked data web by bridging RDF molecule templates. In *Proceedings of the International Conference on Database and Expert Systems Applications (DEXA'17)*. Springer, 3–18.

[46] Ivan Ermilov, Sören Auer, and Claus Stadler. 2013. Csv2rdf: User-driven CSV to RDF mass conversion framework. In *Proceedings of the Conference on Information Systems Engineering and Management (ISEM'13)*, Vol. 13. 4–6.

[47] I. Ermilov, J. Lehmann, M. Martin, and S. Auer. 2016. LODStats: The data web census dataset. In *Proceedings of the International Semantic Web Conference (ISWC'16)*. Springer, 38–46.

[48] Pavlos Fafalios, Thanos Yannakis, and Yannis Tzitzikas. 2016. Querying the web of data with SPARQL-LD. In *International Conference on Theory and Practice of Digital Libraries (TPDL'16)*. Springer, 175–187.

[49] Michael Färber, Frederic Bartscherer, Carsten Menne, and Achim Rettinger. 2018. Linked data quality of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web* 9, 1 (2018), 77–129.

[50] Michael Färber, Basil Ell, Carsten Menne, and Achim Rettinger. 2015. A comparative survey of dbpedia, freebase, opencyc, wikidata, and yago. *Semantic Web J.* 1, 1 (2015), 1–5.

[51] J. D. Fernández, W. Beek, M. A. Martínez-Prieto, and M. Arias. 2017. LOD-a-lot. In *Proceedings of the International Semantic Web Conference (ISWC'17)*. Springer, 75–83.

[52] J. D. Fernández, M. A. Martínez-Prieto, C. Gutiérrez, A. Polleres, and M. Arias. 2013. Binary RDF representation for publication and exchange (HDT). *Web Semantics: Sci. Serv. Agents World Wide Web* 19 (2013), 22–41.

[53] Valeria Fionda, Giuseppe Pirrò, and Claudio Gutierrez. 2015. NautiLOD: A formal language for the web of data graph. *ACM Trans. Web* 9, 1 (2015), 5.
[54] Annika Flemming. 2010. Quality characteristics of linked data publishing datasources. *Master's Thesis, Humboldt-Universität of Berlin* (2010).
[55] Giorgos Flouris, Dimitris Manakanatas, Haridimos Kondylakis, Dimitris Plexousakis, and Grigoris Antoniou. 2008. Ontology change: Classification and survey. *Knowl. Eng. Rev.* 23, 2 (2008), 117–152.
[56] Christian Fürber and Martin Hepp. 2011. Swiqa-a semantic web information quality assessment framework. In *Proceedings of the European Conference on Information Systems (ECIS'11)*, Vol. 15. 19.
[57] D. Gerber, D. Esteves, J. Lehmann, L. Bühmann, R. Usbeck, A. N. Ngomo, and R. Speck. 2015. DeFacto-temporal and multilingual deep fact validation. *Web Semantics: Sci. Serv. Agents World Wide Web* 35 (2015), 85–101.
[58] José M Giménez-Garcıa, Harsh Thakkar, and Antoine Zimmermann. 2016. Assessing trust with pagerank in the web of data. In *Proceedings of the 3rd International Workshop on Dataset PROFIling and fEderated Search for Linked Data*.
[59] Hugh Glaser, Afraz Jaffri, and Ian Millard. 2009. Managing co-reference on the semantic web. (2009).
[60] Olaf Görlitz and Steffen Staab. 2011. Splendid: Sparql endpoint federation exploiting void descriptions. In *Proceedings of the International Conference on Consuming Linked Data (COLD'11)*. 13–24.
[61] P. Groth, A. Loizou, A. J. G. Gray, C. Goble, L. Harland, and S. Pettifer. 2014. API-centric linked data integration: The open PHACTS discovery platform case study. *Web Semantics: Sci. Serv. Agents World Wide Web* 29 (2014), 12–18.
[62] Tobias Grubenmann, Abraham Bernstein, Dmitry Moor, and Sven Seuken. 2017. Challenges of source selection in the WoD. In *Proceedings of the International Semantic Web Conference*. Springer, 313–328.
[63] Christophe Guéret, Paul Groth, Claus Stadler, and Jens Lehmann. 2012. Assessing linked data mappings using network measures. In *The Semantic Web: Research and Applications*. Springer, 87–102.
[64] Yuanbo Guo, Zhengxiang Pan, and Jeff Heflin. 2005. LUBM: A benchmark for OWL knowledge base systems. *Web Semantics: Sci. Serv. Agents World Wide Web* 3, 2–3 (2005), 158–182.
[65] A. Halevy, A. Rajaraman, and J. Ordille. 2006. Data integration: the teenage years. In *Proceedings of the Conference on Very Large Data Bases (VLDB'06)*. 9–16.
[66] Andreas Harth, Craig A. Knoblock, Steffen Stadtmüller, Rudi Studer, and Pedro Szekely. 2013. On-the-fly integration of static and dynamic linked data. In *Proceedings of the International Conference on Consuming Linked Data (COLD'13)*. Citeseer, 1613–0073.
[67] Olaf Hartig. 2009. Provenance information in the web of data. In *Proceedings of the Workshop on Linked Data on the Web (LDOW'09)*.
[68] Olaf Hartig. 2013. An overview on execution strategies for Linked Data queries. *Datenbank-Spektrum* 13, 2 (2013), 89–99.
[69] Olaf Hartig. 2013. SQUIN: A traversal-based query execution system for the web of linked data. In *Proceedings of the ACM Special Interest Group on Management of Data (SIGMOD'13)*. ACM, 1081–1084.
[70] Olaf Hartig and Jun Zhao. 2009. Using web data provenance for quality assessment. In *Proceedings of the International Workshop on the role of Semantic Web in Provenance Management (SWPM'09)*. 29–34.
[71] Olaf Hartig and Jun Zhao. 2010. Publishing and consuming provenance metadata on the web of linked data. In *Provenance and Annotation of Data and Processes*. Springer, 78–90.
[72] A. Hogan, A. Harth, J. Umbrich, S. Kinsella, A. Polleres, and S. Decker. 2011. Searching and browsing linked data with swse: The semantic web search engine. *Web Semantics: Sci. Serv. Agents World Wide Web* 9, 4 (2011), 365–401.
[73] A. Hogan, J. Umbrich, A. Harth, R. Cyganiak, A. Polleres, and S. Decker. 2012. An empirical survey of linked data conformance. *Web Semantics: Sci. Serv. Agents World Wide Web* 14 (2012), 14–44.
[74] K. Hose, R. Schenkel, M. Theobald, and G. Weikum. 2011. Database foundations for scalable RDF processing. In *Proceedings of the International Conference on Reasoning Web: Semantic Technologies for the Web of Data*. Springer-Verlag, 202–249.
[75] Filip Ilievski, Wouter Beek, Marieke van Erp, Laurens Rietveld, and Stefan Schlobach. 2016. LOTUS: Adaptive text search for big linked data. In *Proceedings of the International Semantic Web Conference*. Springer, 470–485.
[76] A. Isaac and B. Haslhofer. 2013. Europeana linked open data—data.europeana.eu. *Semantic Web* 4, 3 (2013), 291–297.
[77] Krzysztof Janowicz, Pascal Hitzler, Benjamin Adams, Dave Kolas, I. I. Vardeman et al. 2014. Five stars of linked data vocabulary use. *Semantic Web* 5, 3 (2014), 173–176.
[78] Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. 2011. Logmap: Logic-based and scalable ontology matching. In *Proceedings of the International Semantic Web Conference*. Springer, 273–288.
[79] Tobias Käfer, Ahmed Abdelrahman, Jürgen Umbrich, Patrick O' Byrne, and Aidan Hogan. 2013. Observing linked data dynamics. In *Proceedings of the Extended Semantic Web Conference*. Springer, 213–227.
[80] Maulik R. Kamdar and Mark A. Musen. 2017. PhLeGrA: Graph analytics in pharmacology over the web of life sciences linked open data. In *Proceedings of the World Wide Web Conference*. 321–329.
[81] Zoi Kaoudi and Ioana Manolescu. 2015. RDF in the clouds: A survey. *VLDB J.* 24, 1 (2015), 67–91.

[82] Michel Klein. 2001. Combining and relating ontologies: An analysis of problems and solutions. In *Proceeding sof the International Joint Conferences on Artificial Intelligence (OIS@ IJCAI'01)*.
[83] T. Knap, J. Michelfeit, J. Daniel, P. Jerman, D. Rychnovskỳ, T. Soukup, and M. Nečaskỳ. 2012. ODCleanStore: A framework for managing and providing integrated linked data on the web. In *Proceedings of the International Conference on Web Information Systems Engineering (WISE'12)*. Springer, 815–816.
[84] Craig A Knoblock and Pedro Szekely. 2015. Exploiting semantics for big data integration. *AI Magazine* 36, 1 (2015).
[85] Haridimos Kondylakis and Dimitris Plexousakis. 2013. Ontology evolution without tears. *Web Semantics: Sci. Serv. Agents World Wide Web* 19 (2013), 42–58.
[86] Dimitris Kontokostas, Patrick Westphal, Sören Auer, Sebastian Hellmann, Jens Lehmann, Roland Cornelissen, and Amrapali Zaveri. 2014. Test-driven evaluation of linked data quality. In *Proceedings of the 23rd World Wide Web Conference (WWW'14)*. ACM, 747–758.
[87] Dimitris Kontokostas, Amrapali Zaveri, Sören Auer, and Jens Lehmann. 2013. *Triplecheckmate: A Tool for Crowdsourcing the Quality Assessment of Linked Data*. Springer, 265–272.
[88] Christina Lantzaki, Panagiotis Papadakos, Anastasia Analyti, and Yannis Tzitzikas. 2017. Radius-aware approximate blank node matching using signatures. *Knowl. Info. Syst.* 50, 2 (2017), 505–542.
[89] Wangchao Le, Songyun Duan, Anastasios Kementsietsidis, Feifei Li, and Min Wang. 2011. Rewriting queries on SPARQL views. In *Proceedings of the 20th International Conference on World Wide Web*. ACM, 655–664.
[90] Jens Lehmann, Robert Isele, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.
[91] Luiz André P. Paes Leme, Giseli Rabello Lopes, Bernardo Pereira Nunes, Marco Antonio Casanova, and Stefan Dietze. 2013. Identifying candidate datasets for data interlinking. In *Proceedings of the International Conference on Web Engineering (ICWE'13)*. Springer, 354–366.
[92] Maurizio Lenzerini. 2002. Data integration: A theoretical perspective. In *Proceedings of the 21st ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*. ACM, 233–246.
[93] Xian Li, Xin Luna Dong, Kenneth Lyons, Weiyi Meng, and Divesh Srivastava. 2012. Truth finding on the deep web: Is the problem solved? In *Proceedings of the VLDB Endowment*, Vol. 6. VLDB Endowment, 97–108.
[94] Vanessa Lopez, Christina Unger, Philipp Cimiano, and Enrico Motta. 2013. Evaluating question answering over linked data. *Web Semantics: Sci. Serv. Agents World Wide Web* 21 (2013), 3–13.
[95] Konstantinos Makris, Nikos Bikakis, Nektarios Gioldasis, and Stavros Christodoulakis. 2012. SPARQL-RW: Transparent query access over mapped RDF data sources. In *Proceedings of the 15th International Conference on Extending Database Technology (EDBT'12)*. ACM, 610–613.
[96] Pablo N. Mendes, Hannes Mühleisen, and Christian Bizer. 2012. Sieve: Linked data quality assessment and fusion. In *Proceedings of the Joint International Conference on Extending Database Technology and International Conference on Database Theory (EDBT/ICDT'12)*. ACM, 116–123.
[97] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *Arxiv Preprint Arxiv:1301.3781*.
[98] N. Minadakis, Y. Marketakis, H. Kondylakis, G. Flouris, M. Theodoridou, M. Doerr, and G. de Jong. 2015. X3ML framework: An effective suite for supporting data mappings. In *Proceedings of the Workshop on Extending, Mapping and Focusing the CRM at the International Conference on Theory and Practice of Digital Libraries (EMF-CRM@ TPDL'15)*. 1–12.
[99] Paolo Missier, Khalid Belhajjame, and James Cheney. 2013. The W3C PROV family of specifications for modelling provenance metadata. In *Proceedings of the 16th International Conference on Extending Database Technology (EDBT'13)*. ACM, 773–776.
[100] Gabriela Montoya. 2016. *Answering SPARQL Queries Using Views*. Ph.D. Dissertation. Université de Nantes.
[101] Gabriela Montoya, Luis-Daniel Ibáñez, Hala Skaf-Molli, Pascal Molli, and Maria-Esther Vidal. 2014. Semlav: Local-as-view mediation for sparql queries. In *Transactions on Large-Scale Data- and Knowledge-Centered Systems XIII*. Springer, 33–58.
[102] Camilo Morales, Diego Collarana, Maria-Esther Vidal, and Sören Auer. 2017. MateTee: A semantic similarity metric based on translation embeddings for knowledge graphs. In *Proceedings of the International Conference on Web Engineering (ICWE'17)*. Springer, 246–263.
[103] L. Moreau, B. Ludäscher, I. Altintas, R. S. Barga, S. Bowers, S. Callahan, G. Chin Jr, B. Clifford et al. 2008. The first provenance challenge. *Concurr. Comput.: Pract. Exper.* 20, 5 (2008), 409–418.
[104] M. Mountantonakis, N. Minadakis, Y. Marketakis, P. Fafalios, and Y. Tzitzikas. 2016. Quantifying the connectivity of a semantic warehouse and understanding its evolution over time. *Int. J. Semantic Web Info. Syst.* 12, 3 (2016), 27–78.
[105] Michalis Mountantonakis and Yannis Tzitzikas. 2016. On measuring the lattice of commonalities among several linked datasets. *Proc. VLDB Endow.* 9, 12 (2016), 1101–1112.

[106] Michalis Mountantonakis and Yannis Tzitzikas. 2018. High performance methods for linked open data connectivity analytics. *Information* 9, 6 (2018).
[107] Michalis Mountantonakis and Yannis Tzitzikas. 2018. Scalable methods for measuring the connectivity and quality of large numbers of linked datasets. *J. Data Info. Qual.* 9, 3 (2018), 15.
[108] Markus Nentwig, Michael Hartung, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm. 2017. A survey of current link discovery frameworks. *Semantic Web* 8, 3 (2017), 419–436.
[109] Markus Nentwig, Tommaso Soru, Axel-Cyrille Ngonga Ngomo, and Erhard Rahm. 2014. LinkLion: A link repository for the web of data. In *Proceedings of the European Semantic Web Conference (ESWC'14)*. Springer, 439–443.
[110] DuyHoa Ngo, Zohra Bellahsene, and R Coletta. 2016. Overview of YAM++—(not) Yet Another Matcher for ontology alignment task. *Web Semantics: Science, Services and Agents on the World Wide Web* 41 (2016), 30–49.
[111] Axel-Cyrille Ngonga Ngomo and Sören Auer. 2011. Limes-a time-efficient approach for large-scale link discovery on the web of data. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI'11)*. 2312–2317.
[112] Andriy Nikolov, Mathieu d' Aquin, and Enrico Motta. 2011. What should i link to? identifying relevant sources and classes for data linking. In *Proceedings of the Joint International Semantic Technology Conference*. Springer, 284–299.
[113] N. F. Noy. 2004. Semantic integration: A survey of ontology-based approaches. *ACM SIGMOD Rec.* 33, 4 (2004), 65–70.
[114] Peter Ochieng and Swaib Kyanda. 2018. A statistically based ontology matching tool. *Distrib. Parallel Databases* 36, 1 (2018), 195–217.
[115] D. Oguz, B. Ergenc, S. Yin, O. Dikenelli, and A. Hameurlain. 2015. Federated query processing on linked data: A qualitative survey and open challenges. *Knowl. Eng. Rev.* 30, 5 (2015), 545–563.
[116] Eyal Oren, Renaud Delbru, Michele Catasta, Richard Cyganiak, Holger Stenzhorn, and Giovanni Tummarello. 2008. Sindice.com: A document-oriented lookup index for open linked data. *Int. J. Metadata Semantics Ontol.* 3, 1 (2008), 37–52.
[117] Lorena Otero-Cerdeira, Francisco J. Rodríguez-Martínez, and Alma Gómez-Rodríguez. 2015. Ontology matching: A literature review. *Expert Syst. Appl.* 42, 2 (2015), 949–971.
[118] Jeff Pasternack and Dan Roth. 2013. Latent credibility analysis. In *Proceedings of the World Wide Web Conference (WWW'13)*. ACM, 1009–1020.
[119] Heiko Paulheim. 2017. Knowledge graph refinement: A survey of approaches and evaluation methods. *Semantic Web* 8, 3 (2017), 489–508.
[120] Peng Peng, Lei Zou, M Tamer Özsu, Lei Chen, and Dongyan Zhao. 2016. Processing SPARQL queries over distributed RDF graphs. *VLDB J.* 25, 2 (2016), 243–268.
[121] Petar Petrovski, Volha Bryl, and Christian Bizer. 2014. Integrating product data from websites offering microdata markup. In *Proceedings of the 23rd International Conference on World Wide Web*. ACM, 1299–1304.
[122] B. Quilitz and U. Leser. 2008. Querying distributed RDF data sources with SPARQL. In *Proceedings of the European Semantic Web Conference (ESWC'08)*. Springer, 524–538.
[123] Erhard Rahm. 2016. The Case for holistic data integration. In *Proceedings of the East European Conference on Advances in Databases and Information Systems (ADBIS'16)*. Springer, 11–27.
[124] Thomas Rebele, Fabian Suchanek, Johannes Hoffart, Joanna Biega, Erdal Kuzey, and Gerhard Weikum. 2016. YAGO: A multilingual knowledge base from wikipedia, wordnet, and geonames. In *Proceedings of the International Semantic Web Conference (ISWC'16)*. Springer, 177–185.
[125] Theodoros Rekatsinas, Xin Luna Dong, Lise Getoor, and Divesh Srivastava. 2015. Finding quality in quantity: The challenge of discovering valuable sources for integration. In *Proceedings of the Conference on Innovative Data Systems Research (CIDR'15)*.
[126] L. Rietveld, W. Beek, and S. Schlobach. 2015. LOD lab: Experiments at LOD scale. In *Proceedings of the International Semantic Web Conference (ISWC'15)*. Springer, 339–355.
[127] Petar Ristoski, Christian Bizer, and Heiko Paulheim. 2015. Mining the web of linked data with rapidminer. *Web Semantics: Sci. Serv. Agents World Wide Web* 35 (2015), 142–151.
[128] P. Ristoski and H. Paulheim. 2016. Rdf2vec: Rdf graph embeddings for data mining. In *Proceedings of the International Semantic Web Conference (ISWC'16)*. Springer, 498–514.
[129] Muhammad Saleem, Ali Hasnain, and Axel-Cyrille Ngonga Ngomo. 2018. Largerdfbench: A billion triples benchmark for sparql endpoint federation. *J. Web Semantics* (2018).
[130] Muhammad Saleem, Yasar Khan, Ali Hasnain, Ivan Ermilov, and Axel-Cyrille Ngonga Ngomo. 2016. A fine-grained evaluation of SPARQL endpoint federation systems. *Semantic Web* 7, 5 (2016), 493–518.
[131] Muhammad Saleem and Axel-Cyrille Ngonga Ngomo. 2014. Hibiscus: Hypergraph-based source selection for sparql endpoint federation. In *Proceedings of the European Semantic Web Conference*. Springer, 176–191.
[132] Muhammad Saleem, Axel-Cyrille Ngonga Ngomo, Josiane Xavier Parreira, Helena F. Deus, and Manfred Hauswirth. 2013. Daw: Duplicate-aware federated query processing over the web of data. In *Proceedings of the International Semantic Web Conference (ISWC'13)*. Springer, 574–590.

[133] M. Saleem, S. Padmanabhuni, A. N. Ngomo, A. Iqbal, J. S. Almeida, S. Decker, and H. F. Deus. 2014. TopFed: TCGA tailored federated query processing and linking to LOD. *J. Biomed. Semantics* 5, 1 (2014), 1.

[134] Cristina Sarasua, Elena Simperl, and Natalya F. Noy. 2012. Crowdmap: Crowdsourcing ontology alignment with microtasks. In *Proceedings of the International Semantic Web Conference*. Springer, 525–541.

[135] Cristina Sarasua, Steffen Staab, and Matthias Thimm. 2017. Methods for intrinsic evaluation of links in the web of data. In *Proceedings of the European Semantic Web Conference*. Springer, 68–84.

[136] Max Schmachtenberg, Christian Bizer, and Heiko Paulheim. 2014. Adoption of the linked data best practices in different topical domains. In *Proceedings of the International Semantic Web Conference (ISWC'14)*. Springer, 245–260.

[137] Michael Schmidt, Olaf Görlitz, Peter Haase, Günter Ladwig, Andreas Schwarte, and Thanh Tran. 2011. Fedbench: A benchmark suite for federated semantic data query processing. In *Proceedings of the International Semantic Web Conference (ISWC'11)*. 585–600.

[138] Falk Scholer, Diane Kelly, and Ben Carterette. 2016. Information retrieval evaluation using test collections. *Info. Retrieval J.* 19, 3 (2016), 225–229.

[139] Andreas Schultz, Andrea Matteini, Robert Isele, Pablo N. Mendes, Christian Bizer, and Christian Becker. 2012. LDIF-a framework for large-scale Linked Data integration. In *Proceedings of the 21st International World Wide Web Conference, Developers Track (WWW'12)*.

[140] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. 2011. Fedx: Optimization techniques for federated query processing on linked data. In *Proceedings of the International Semantic Web Conference*. Springer, 601–616.

[141] Juan F. Sequeda, Marcelo Arenas, and Daniel P. Miranker. 2012. On directly mapping relational databases to RDF and OWL. In *Proceedings of the 21st International Conference on World Wide Web*. ACM, 649–658.

[142] Juan F. Sequeda, Syed Hamid Tirmizi, Oscar Corcho, and Daniel P. Miranker. 2011. Survey of directly mapping sql databases to the semantic web. *Knowl. Eng. Rev.* 26, 4 (2011), 445–486.

[143] Nigel Shadbolt, Tim Berners-Lee, and Wendy Hall. 2006. The semantic web revisited. *IEEE Intell. Syst.* 21, 3 (2006), 96–101.

[144] Pavel Shvaiko and Jérôme Euzenat. 2013. Ontology matching: State of the art and future challenges. *IEEE Trans. Knowl. Data Eng.* 25, 1 (2013), 158–176.

[145] Karen Smith-Yoshimura. 2018. Analysis of 2018 international linked data survey for implementers. *Code* {4} *lib J.* 42 (2018).

[146] Stefano Spaccapietra and Christine Parent. 1994. View integration: A step forward in solving structural conflicts. *IEEE Trans. Knowl. Data Eng.* 6, 2 (1994), 258–274.

[147] Stefano Spaccapietra, Christine Parent, and Yann Dupont. 1992. Model independent assertions for integration of heterogeneous schemas. *VLDB J.* 1, 1 (1992), 81–126.

[148] Fabian M. Suchanek, Serge Abiteboul, and Pierre Senellart. 2011. Paris: Probabilistic alignment of relations, instances, and schema. *Proc. VLDB Endow.* 5, 3 (2011), 157–168.

[149] Ignacio Traverso, Maria-Esther Vidal, Benedikt Kämpgen, and York Sure-Vetter. 2016. GADES: A graph-based semantic similarity measure. In *Proceedings of the 12th International Conference on Semantic Systems*. ACM, 101–104.

[150] Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. Lc-quad: A corpus for complex question answering over knowledge graphs. In *Proceedings of the International Semantic Web Conference*. Springer, 210–218.

[151] Yannis Tzitzikas et al. 2013. Integrating heterogeneous and distributed information about marine species through a top level ontology. In *Proceedings of the International Conference on Metadata and Semantics Research (MTSR'13)*. Springer, 289–301.

[152] Yannis Tzitzikas, Mary Kampouraki, and Anastasia Analyti. 2013. Curating the specificity of ontological descriptions under ontology evolution. *J. Data Semantics* (2013), 1–32.

[153] Yannis Tzitzikas, Nikos Manolis, and Panagiotis Papadakos. 2017. Faceted exploration of RDF/S datasets: A survey. *J. Intell. Info. Syst.* 48, 2 (2017), 329–364.

[154] Yannis Tzitzikas, Yannis Marketakis et al. 2017. Towards a global record of stocks and fisheries. In *Proceedings of the 8th International HAICTA Conference*. 328–340.

[155] Yannis Tzitzikas and Carlo Meghini. 2003. Ostensive automatic schema mapping for taxonomy-based peer-to-peer systems. In *Proceedings of the International Workshop on Cooperative Information Agents*. Springer, 78–92.

[156] Y. Tzitzikas, N. Minadakis, Y. Marketakis, P. Fafalios, C. Allocca, M. Mountantonakis, and I. Zidianaki. 2014. Matware: Constructing and exploiting domain specific warehouses by aggregating semantic data. In *Proceedings of the European Semantic Web Conference (ESWC'14)*. Springer, 721–736.

[157] Yannis Tzitzikas, Nicolas Spyratos, and Panos Constantopoulos. 2005. Mediators over taxonomy-based information sources. *VLDB J.* 14, 1 (2005), 112–136.

[158] J. Urbani, S. Kotoulas, J. Maassen, F. Van Harmelen, and H. Bal. 2012. WebPIE: A web-scale parallel inference engine using mapreduce. *Web Semantics: Sci. Serv. Agents World Wide Web* 10 (2012), 59–75.

[159] Pierre-Yves Vandenbussche, Jürgen Umbrich, Luca Matteis, Aidan Hogan, and Carlos Buil-Aranda. 2017. SPARQLES: Monitoring public SPARQL endpoints. *Semantic Web* 8, 6 (2017), 1049–1065.

[160] Panos Vassiliadis and Timos Sellis. 1999. A survey of logical models for OLAP databases. *ACM SIGMOD Rec.* 28, 4 (1999), 64–69.

[161] Julius Volz, Christian Bizer, Martin Gaedke, and Georgi Kobilarov. 2009. Silk—A link discovery framework for the web of data. In *Proceedings of the World Wide Web Workshop on Linked Data on the Web*.

[162] D. Vrandečić and M. Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM* 57, 10 (2014), 78–85.

[163] Andreas Wagner, Peter Haase, Achim Rettinger, and Holger Lamm. 2014. Entity-based data source contextualization for searching the web of data. In *Proceedings of the European Semantic Web Conference*. Springer, 25–41.

[164] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg et al. 2016. The FAIR Guiding principles for scientific data management and stewardship. *Sci. Data* 3 (2016).

[165] Ramana Yerneni, Chen Li, Jeffrey Ullman, and Hector Garcia-Molina. 1999. Optimizing large join queries in mediation systems. In *Proceedings of the 7th International Conference on Database Theory (ICDT'99)*. 348–364. http://dl.acm.org/citation.cfm?id=645503.656258.

[166] S. Yumusak, E. Dogdu, H. Kodaz, A. Kamilaris, and P. Vandenbussche. 2017. SpEnD: Linked data SPARQL endpoints discovery using search engines. *IEICE Trans. Info. Syst.* 100, 4 (2017), 758–767.

[167] Amrapali Zaveri, Dimitris Kontokostas, Mohamed A. Sherif, Lorenz Bühmann, Mohamed Morsey, Sören Auer, and Jens Lehmann. 2013. User-driven quality evaluation of dbpedia. In *Proceedings of the European Conference on Semantic Technologies and Artificial Intelligence (SEMANTiCS'13)*. ACM, 97–104.

[168] Amrapali Zaveri, Anisa Rula, Andrea Maurino, Ricardo Pietrobon, Jens Lehmann, and Sören Auer. 2015. Quality assessment for linked data: A survey. *Semantic Web* 7, 1 (2015), 63–93.

[169] Linhong Zhu, Majid Ghasemi-Gol, Pedro Szekely, Aram Galstyan, and Craig A. Knoblock. 2016. Unsupervised entity resolution on multi-type graphs. In *Proceedings of the International Semantic Web Conference*. Springer, 649–667.

Received July 2018; revised May 2019; accepted July 2019