On Learnability of Constraints from RDF Data

Emir Muñoz[1,2(✉)]

[1] Fujitsu Ireland Limited, Dublin, Ireland
[2] Insight Centre for Data Analytics, National University of Ireland, Galway, Ireland
emir.munoz@insight-centre.org

Abstract. RDF is structured, dynamic, and schemaless data, which enables a big deal of flexibility for Linked Data to be available in an open environment such as the Web. However, for RDF data, flexibility turns out to be the source of many data quality and knowledge representation issues. Tasks such as assessing data quality in RDF require a different set of techniques and tools compared to other data models. Furthermore, since the use of existing schema, ontology and constraint languages is not mandatory, there is always room for misunderstanding the structure of the data. Neglecting this problem can represent a threat to the widespread use and adoption of RDF and Linked Data. Users should be able to *learn* the characteristics of RDF data in order to determine its fitness for a given use case, for example. For that purpose, in this doctoral research, we propose the use of constraints to inform users about characteristics that RDF data naturally exhibits, in cases where ontologies (or any other form of explicitly given constraints or schemata) are not present or not expressive enough. We aim to address the problems of defining and discovering classes of constraints to help users in data analysis and assessment of RDF and Linked Data quality.

Keywords: RDF constraints · Linked data mining · Data quality · Data semantics

1 Introduction

Background. The flexibility of RDF comes from the Web Ontology Language (OWL): Open World Assumption (OWA), missing information treated as unknown; and Unique Name Assumption (UNA), individuals may have more than one name. These characteristics difficult data validation [28] and data quality assessment. The notion of Data Quality (DQ) is related to individual use cases and cannot be assessed independently from the user. Yet there is a lack of methodologies tailored to Linked Data (LD) that consider users' requirements and users without previous experience on the data. In this doctoral research, we explore the use of constraints as a tool for users to identify the modeling behind data represented using RDF. Constraints are limitations incorporated on the data that are supposed to be satisfied all the time by instances of the

DOI: 10.1007/978-3-319-34129-3_52

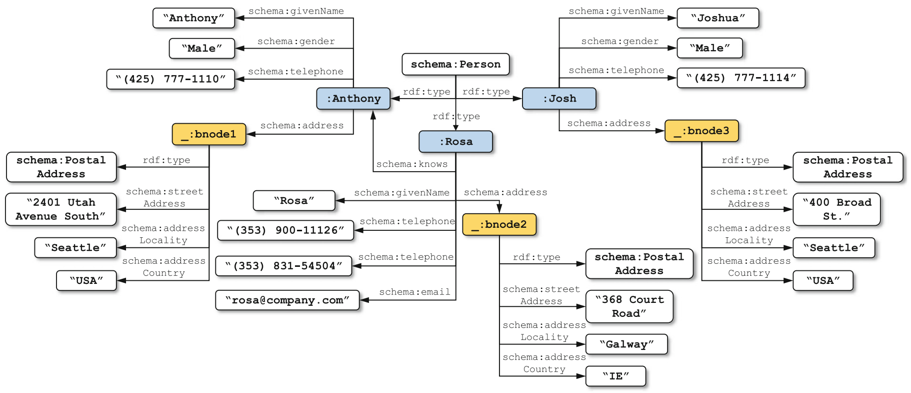

Fig. 1. RDF data describing information about people.

database [1]. They are useful for users to understand data as they represent characteristics that data naturally exhibits [15]. However, a deeper study of constraints and the benefits that they bring to the RDF model, especially for use cases like quality assessment, has not yet been made.

Motivation. As of April 2014, an amount of 1014 Linked Datasets were registered in the Linked Open Data (LOD) cloud [19], containing billions of RDF statements. Due to the heterogeneity of the schema(s) and modeling used in each one of the sources, the task of determining whether a dataset is relevant for a given use case becomes a daunting and non-trivial problem. A way to understand data is by understanding the model behind that all facts follow. However, under the OWA multiple possible models can be satisfied by the facts. Considering the data as complete can allow us to adopt a Closed World Assumption (OWA) with UNA (sacrificing flexibility), which is equivalent to a relational database [16,28], where a single model contains all and only the facts assessed. Although advances in the Semantic Web and Linked Data have been made in standards to communicate the semantics behind data (*e.g.*, ontologies), they most of the time fall short in providing expressive schemas. Therefore, there is a clear need for methods and tools to help users to comprehend the structure of RDF data in the presence of inconsistencies and poor or absent schema.

The problem of lacking structure of RDF data can be addressed by extracting constraints. In the RDF model, they can help to cope with this lack of schema, representing some missing meta-data, such as identifiers, and cardinalities. Though we sacrifice some of the greater flexibility and interoperability.

Example 1. *In the following we consider the RDF data in Fig. 1, which contains information about people. Each entity is defined as a member of the class* `schema:Person`*, thus they can have all properties of this class. We can describe some characteristics of the data (in Fig. 1) by means of constraints:*

C1. Every person contains exactly one value for the schema:givenName *and* schema:address *properties.*

C2. The combined properties schema:givenName *and* schema:address *uniquely identify each person in the data, every person has a different pair of values.*

C3. Each person is connected to at least one value for the schema:telephone *property, and at most two values.*

C4. The value of the property schema:knows *in the person* :Rosa *makes direct reference to the person* :Anthony*.*

C5. The property schema:email *is uncommon in the data and, in this dataset, has a probability of* $\frac{1}{3}$ *to appear once.*

C6. All values of the property schema:telephone *follow the same '(NUMBER) NUMBER-NUMBER' syntactic pattern.*

C7. Entities with a schema:givenName *and* schema:address *must be instances of the class* schema:Person*.*

C8. Values for property schema:gender *are expected to be of type text (string).*

By looking at the constraints in the example, we can see that they partially uncover the structure and schema of the data. Formally, constraints *C2* and *C4* are known as *integrity constraints* (used to ensure accuracy and consistency of data), whilst constraints *C1* and *C3* are known as *cardinality constraints* (used to specify a minimum and maximum bound for relationships with properties that an entity can have [15,29]). *C5* is special in the sense it expresses the marginal probability by which the constraint holds in the dataset [4,6]. *C6* is known as *syntactic pattern constraint* [17], and restrict the syntax that the values of a property can have. Constraints *C7* and *C8* are known as *domain and range constraints*, respectively; they restrict the types that entities (in the domain) and values (in the range) of relations for a given property can have, respectively.

In databases, applications of constraints include: *data cleaning*, constraints verify that the data conforms to a basic level of data consistency and correctness, preventing the introduction of dirty data; *integration*, *modeling*, *retrieval*, among others [1]. Although other classes of constraints can be defined for RDF, in this work, we will limit ourselves to mainly study the ones described above because of: (a) their potential for DQ analysis; and (b) the lack of existing research on learning such constraints from the data. Most of the actions involved in LD quality assessment, in order to be performed, require users to have knowledge on the underlying structure of the data. Here, we aim to provide such a notion to the users in the absence of explicit programming, by means of identifying constraints in the data. In this thesis work, we refer to this problem as *"learnability of constraints"*. During this research our goal is three-fold: (1) increase the expressiveness of RDF constraints to enrich the user's understanding of the RDF data modeling; (2) present algorithms to automatically extract such constraints; and (3) study their effects during the assessment of quality in RDF data.

2 State of the Art

The concept of constraints has been present for long time in databases [1], and only recently has been introduced in RDF [14]. Lausen et al. [14] presents an approach for translating constraints while converting a relational database to RDF without losing semantic information. The authors extended the RDF vocabulary in order to encode integrity constraints such as keys and foreign keys. Theoretical aspects of integrity constraints in RDF and RDFS were studied in [2,7], introducing a mechanism to express functional constraints in terms of equality. While constraints like keys are indispensable for data consistency in relational databases, in the RDF model they are still not considered first-class citizens.

OWL 2[1] is the most straightforward way to represent some of the constraint here introduced. Key constraints can be defined using the axiom `owl:hasKey` based upon object and data properties. Cardinality constraints can also be expressed in OWL 2 by means of three expressions: `owl:minCardinality`, `owl:maxCardinality`, and `owl:exactCardinality`. However, the semantics of OWL 2 considers OWA and does not make the UNA, making it hard to evaluate consistency when same or different individual relations are not explicitly stated (very common in lightweight ontologies). Tao *et al.* [28] also shows that the expressiveness allowed by OWL 2 is limited to express Integrity Constraints.

Due to the huge effort involved in generating full ontologies, several works aim to (semi)automatically learn or enrich ontologies from text or instance data. Property domain/range constraints have been studied in [30], and used to acquire class disjointness axioms to detect inconsistencies in DBpedia. Völker et al. [31] uses two approaches for the same problem: extensional (relying solely on instances) and intensional (based on logical and lexical description of classes). Key constraints have been explored by [23,27] for link discovery and data integration purposes. And our previous work [17] introduces the concept of syntactic constraints in RDF. (For a recent survey on methods in this line see [18].)

Languages for expressing constraints in RDF data intended to validate RDF documents, and communicate expected patterns exists. Among the most populars is Shape Expressions (ShEx)[2], which is more focused on type inference than on verification, unlike the RDF Data Descriptions (RDD) [20]. RDD uses a compact special-purpose syntax which is independent of a specific inference machinery. Shapes Constraint Language (SHACL)[3] is the recent output of the W3C RDF Data Shapes Working Group, which provides a high-level vocabulary to identify predicates and their associated cardinalities, datatypes and other constraints (by using SPARQL). SPARQL Infering Notation (SPIN)[4] is a low-level language that allows users to use SPARQL to specify rules and logical constraints. All of them, namely, ShEx, RDD, SHACL, and SPIN, are aimed to validate RDF data, and communicate data semantics among users. They do cover constraints such as keys and cardinality; however, their expressivity is limited compared to our proposal and they do not consider a probabilistic notion.

Multiple applications of constraints can be envisioned in RDF based on their success in the relational model, including: data cleaning, integration, modeling,

[1] https://www.w3.org/TR/owl2-syntax/.
[2] https://www.w3.org/2013/ShEx/Primer.
[3] https://www.w3.org/TR/shacl/.
[4] http://spinrdf.org/.

processing, and retrieval [21]. In terms of quality assurance, keys were recently used in [24] to determine discriminability of resources, *i.e.*, determine if datasets contain indistinguishable resources *w.r.t.* a given set of properties. Likewise, we plan the application of constraints as a tool that serves users to understand the structure of the data and help them to assess RDF data quality.

The discovery task of constraints from RDF data is a new topic and has been focused solely on keys and under a particular and limited view of RDF, which is known as *Concise Bounded Description* (CBD)[5]. CBD limits the (over)use of blank nodes (*e.g.*, the ones generated in the `schema:address` property shown in Example 1), the path length, total number of statements, and reifications. It also adds an unexpected complexity to the discovery problem.

3 Problem Statement and Contributions

The benefits that the definition of expressive classes of constraints for RDF data brings to the table are manifold, *e.g.*, data cleaning, integration, modeling, processing, and retrieval, akin to constraints in relational databases. However, they have not been studied in depth by the Linked Data community. Although various of the existing technologies can be used to define new classes of constraints, there is no practical or theoretical framework that connects them yet. In this Ph.D. research, we study solutions to implement constraints in RDF using current theoretical foundations. We also aim to show how users can benefit from the presence of constraints while assessing quality of RDF datasets.

Unlike previous works on key constraints and linked data quality, we acknowledge the volume, variety, veracity, and velocity of data. Therefore, we rely on the following assumptions: (i) we do not assume the existence of a full/complete/rich schema or ontology; (ii) RDF data is not always formatted in a specific way (*e.g.*, CBD), this is more unlikely if we consider the Web of Data; (iii) RDF data contain plenty of RDF Blank Nodes [12,13], which radically change the current view of constraints solely based on *sets of properties* to a view of *sets of property paths*; and (iv) we assume that RDF data are error prone, and incomplete, which requires that these techniques work under uncertainty. The problem and assumptions stated above lead to the following research questions:

RQ1. *Can we define more expressive and novel constraints for RDF data?*
We observed that RDF key constraints are defined as a set of single properties and under the assumption of CBD [25]. We thus see three drawbacks of this approach: (a) CBD is application dependent and complex to compute, and (b) it does not consider complex values (*e.g.*, `schema:address` in Example 1), (c) it does not take into account that RDF data on the Web of Data contains plenty of blank nodes [12,13]. For example, considering Example 1 and the state-of-the-art, we have that constraint *C5* cannot be expressed with current approaches; and, constraint *C2* cannot be expressed under the CBD assumption, and under the assumption

[5] https://www.w3.org/Submission/CBD/.

that two blank nodes are always considered to be equal. Nevertheless, a relaxed version of *C2* including only the `schema:givenName` property can be expressed with current approaches. Here, we aim to increase the coverage and expressivity of RDF constraints with the so-called SPARQL Property Paths [22] and a value similarity definition that can help to cope with complex values.

RQ2. *Can constraints be automatically extracted under a non-CBD assumption?* So far the task of defining constraints for RDF data has been relegated to users mainly with the help of high/low-level languages. ROCKER [23] was the first machine-learning-based approach for key discovery. We expect to determine whether the methods behind ROCKER can be extended to identify more expressive keys and our new classes of constraints. Regardless of the solution, we aim to account the existence of blank nodes, and to consider scalability as a key feature of our framework.

RQ3. *What is the impact of constraints in the assessment of RDF data quality?* Constraints in RDF have been used for data validation mainly. Specific applications of RDF keys have been investigated in data linkage by [5,23,27]. But the semantics of constraints can have several implications in data quality assessment that have not been investigated yet. In [24] the authors present an approach to discover redundant entities in RDF data using a key discovery algorithm. However, we want to investigate further how to exploit the rich semantics of constraints in RDF, especially considering different DQ scenarios and dimensions.

4 Research Methodology and Approach

The expressivity of current RDF constraints definitions is limited, and do not deal with complex data values. Currently, the definition of constraints is based only on sets of properties and does not consider the graph structure of RDF data. This is partially due to certain intractabilities when working with complex data values. For instance, to determine whether two blank nodes are equal (*e.g.*, `_:bnode1` and `_:bnode3` in Example 1), it is needed to determine if there exists an isomorphism between the RDF graphs — which results to be an NP (GI-complete) problem [12]. In order to cope with these limitations, our proposal unfolds in the following subsections.

4.1 Definition of Constraints for RDF

The semantics of constraints in Example 1 is similar across different vocabularies, and similar to their pairs in relational databases. However, current vocabularies/ontologies do not consider complex data values or blank nodes in the constraints definition. In order to cope with this lack of expressivity, our approach includes the definition of constraints using existing standards like SPARQL property paths [22]. In turn, an RDF key could be defined as a set of property paths, instead of single properties. For example, we could express that the path `schema:address/schema:streetAddress` is used as a key. This will also require us to deal with blank nodes and their equivalence. For this we plan to use the skolemization algorithm proposed by Hogan [12].

RDF data may also contain uncertain data, which usually lead to inconsistencies in later processing. In these cases, a strict consideration of constraints would lead to data loss. Allowing some exceptions can prevent applications from losing data [11]. For that we believe that it is missing a definition of RDF constraints with an associated probability of occurrence. Constraint *C5* in Example 1 is an example of this case.

4.2 Discovery of Constraints

Together with the definition of more expressive classes of constraints, we will propose algorithms to discover them from RDF data. Existing approaches only discover key constraints from RDF data in a CBD form, and domain/range constraints. Here, we can follow two non-exclusive possible paths: (i) determine whether existing approaches, such as ROCKER or SAKey, can be extended to identify more expressive keys, and if they are suitable for other classes of constraints (*e.g.*, cardinality, probabilistic, syntactic); (ii) analyse the translation of approaches used to extract constraints in other data models, for example, the XML model [3,10]. (XML is meant for data serialization as a tree where order is important, RDF is a knowledge graph and order is not important.)

4.3 Constraints and Data Quality

We argue that constraints will allow users to efficiently understand the structure and nature of the data. This empowers users to perform different analytic tasks, being DQ one of them. Constraints could be related with several of the DQ dimensions defined in [26,32], such as, relevancy (data helps to know what you want), completeness (data do not leave any open questions), amount of data, interpretability, concise representation, consistent representation, to name just a few. These dimensions are categorized as contextual or representational DQ characteristics of high-quality data [26]. Constraints as the ones listed in Example 1 could be defined and extracted from RDF data to help users in the identification and analysis of those dimensions. A more practical study on how users can benefit from these constraints should be done to derive relations between constraints and DQ dimensions.

5 Preliminary Results

Our first step into the path of learning the structure of RDF data was at property values level. In [17] we provide an unsupervised approach to extract syntactic patterns from property values used in RDF data. These patterns attempt to address the lack of `rdfs:range` definitions. A set of patterns extracted for a given property enables: (i) human-understanding of syntactic patterns that each property follows, (ii) a structural description of properties, (iii) the detection of data inconsistencies, and (iv) the validation and suggestion of new values for a property. In the learning process, first, a lexical analysis is applied over values of properties to generate a sequence of tokens. From these tokens, lexico-syntactic rules are extracted to generate content patterns. For instance, a pattern for the values of property `schema:telephone` in Example 1 could be as `'(NUMBER) NUMBER-NUMBER'`. We experimented using DBpedia v3.9 to extract content patterns for all properties in the ontology. The extraction generated a database with *ca.* 500,000 content patterns.

The extraction of content patterns could be used to enrich the definition of `rdfs:range` properties. Interestingly, a content pattern goes beyond the current definition of a RangeTypeConstraint in RDD, which is limited to indicate that a property points either a URI, BlankNode, Resource, or a Literal.

6 Evaluation Plan

In order to evaluate the outputs of this Ph.D. research, we plan the following segmented evaluation, for the three main parts of this work.

6.1 Definition of Constraints for RDF

A definition for the types of constraints we propose here can be evaluated in terms of the expressivity and applicability. The expressivity of our definition of RDF keys can be compared against OWL 2, ShEx, SHACL, RDD, and [23]. While for the rest of our definitions, to the best of our knowledge, there is no work to compare with. Conversely, we plan to use related works performed in XML [8,9] and relational databases [6] to define semantically similar constraints and applications.

6.2 Discovery of Constraints

The quality of the discovery part can be evaluated against evaluation datasets, measuring the number of retrieved constraints, the time and memory complexity, precision, recall, correctness, and completeness. In the state-of-the-art, we could find only the datasets used in [23], which can be used to partially evaluate our extracted RDF keys. Since there is no other work that considers SPARQL property paths nor probabilistic constraints, there is a lack of manually-annotated gold standards that we can use. We will generate new evaluation datasets to show the effectiveness of our approach, considering different data sources such as Web Data Commons[6]. In order to measure scalability, these datasets must be of different sizes (from thousands up to millions).

6.3 Constraints and Data Quality

The evaluation of this interaction between constraints and data quality will be measured in two ways. First, we can express our constraints in ShEx or RDD, and apply existing tools to validate a dataset against a set of constraints. Second, we plan to perform a user study to determine how useful are the extracted constraints for several scenarios and tasks of quality assessment.

[6] http://webdatacommons.org/.

7 Conclusions and Future Work

From the beginning, the definition of constraints in RDF has been limited by their mapping from relational databases. Current constraints are based on single property names to, for example, uniquely identify entities in a dataset. Thereby, current approaches do not consider the existence of complex property values. Therefore there is a lack of expressiveness and applications for RDF constraints.

In this paper, we present a doctoral research proposal that identifies research questions in the area of constraints for RDF and data quality. This research proposes the definition of more expressive classes of RDF constraints, and the development of methodologies where constraints can help users to understand the structure behind RDF data. We believe this kind of tools will help users in the assessment of quality. Also it will unlock further applications in data cleaning, integration, modeling, processing, and retrieval, akin to constraints in relational databases.

Acknowledgments. This thesis is supervised by Dr. Matthias Nickles. The author would like to thank Prof. Dr. Heiko Paulheim for his valuable comments and suggestions. The work presented in this paper has been supported by TOMOE project funded by Fujitsu Laboratories Limited and Insight Centre for Data Analytics at NUI Galway.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases: The Logical Level, 1st edn. Addison-Wesley, Boston (1995)
2. Akhtar, W., Cortés-Calabuig, A., Paredaens, J.: Constraints in RDF. In: 4th International Workshops on Semantics in Data and Knowledge Bases, SDKB, pp. 23–39 (2010)
3. Arenas, M., Daenen, J., Neven, F., Ugarte, M., den Bussche, J.V., Vansummeren, S.: Discovering XSD keys from XML data. ACM Trans. Database Syst. **39**(4), 28:1–28:49 (2014)
4. Atencia, M., et al.: Defining key semantics for the RDF datasets: experiments and evaluations. In: Hernandez, N., Jäschke, R., Croitoru, M. (eds.) ICCS 2014. LNCS, vol. 8577, pp. 65–78. Springer, Heidelberg (2014)
5. Atencia, M., David, J., Scharffe, F.: Keys and pseudo-keys detection for web datasets cleansing and interlinking. In: ten Teije, A., Völker, J., Handschuh, S., Stuckenschmidt, H., d'Acquin, M., Nikolov, A., Aussenac-Gilles, N., Hernandez, N. (eds.) EKAW 2012. LNCS, vol. 7603, pp. 144–153. Springer, Heidelberg (2012)
6. Brown, P., Link, S.: Probabilistic keys for data quality management. In: Zdravkovic, J., Kirikova, M., Johannesson, P. (eds.) CAiSE 2015. LNCS, vol. 9097, pp. 118–132. Springer, Heidelberg (2015)
7. Cortés-Calabuig, A., Paredaens, J.: Semantics of constraints in RDFS. In: Proceedings of the 6th Alberto Mendelzon International Workshop on Foundations of Data Management, pp. 75–90 (2012)

8. Ferrarotti, F., Hartmann, S., Link, S., Marin, M., Muñoz, E.: The finite implication problem for expressive XML keys: foundations, applications, and performance evaluation. In: Hameurlain, A., Küng, J., Wagner, R., Liddle, S.W., Schewe, K.-D., Zhou, X. (eds.) TLDKS X. LNCS, vol. 8220, pp. 60–94. Springer, Heidelberg (2013)
9. Ferrarotti, F., Hartmann, S., Link, S., Marin, M., Muñoz, E.: Soft cardinality constraints on XML data. In: Lin, X., Manolopoulos, Y., Srivastava, D., Huang, G. (eds.) WISE 2013, Part I. LNCS, vol. 8180, pp. 382–395. Springer, Heidelberg (2013)
10. Grahne, G., Zhu, J.: Discovering approximate keys in XML data. In: Proceedings of the 2002 ACM CIKM, pp. 453–460 (2002)
11. Hartmann, S.: Soft constraints and heuristic constraint correction in entity-relationship modelling. In: Bertossi, L., Katona, G.O.H., Schewe, K.-D., Thalheim, B. (eds.) Semantics in Databases 2001. LNCS, vol. 2582, pp. 82–99. Springer, Heidelberg (2003)
12. Hogan, A.: Skolemising blank nodes while preserving isomorphism. In: Proceedings of the 24th WWW, pp. 430–440 (2015)
13. Hogan, A., Arenas, M., Mallea, A., Polleres, A.: Everything you always wanted to know about blank nodes. Web Semant. Sci. Serv. Agents World Wide Web **27–28**, 42–69 (2014). Semantic Web Challenge 2013
14. Lausen, G., Meier, M., Schmidt, M.: SPARQLing constraints for RDF. In: Proceeding of the 11th EDBT, pp. 499–509 (2008)
15. Liddle, S.W., Embley, D.W., Woodfield, S.N.: Cardinality constraints in semantic data models. Data Knowl. Eng. **11**(3), 235–270 (1993)
16. Motik, B., Horrocks, I., Sattler, U.: Bridging the gap between OWL and relational databases. Web Semant. Sci. Serv. Agents World Wide Web **7**(2), 74–89 (2009)
17. Muñoz, E.: Learning content patterns from linked data. In: Proceedings of the Linked Data for Information Extraction (LD4IE) Workshop, ISWC, CEUR Workshop Proceedings, vol. 1267, pp. 21–32. CEUR-WS.org (2014)
18. Paulheim, H.: Knowledge graph refinement: a survey of approaches and evaluation methods. Semant. Web - Interoperability Usability Appl. IOS Press J. (2016, to appear). http://www.semantic-web-journal.net/
19. Schmachtenberg, M., Bizer, C., Paulheim, H.: Adoption of the linked data best practices in different topical domains. In: Mika, P., et al. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 245–260. Springer, Heidelberg (2014)
20. Schmidt, M., Lausen, G.: Pleasantly consuming linked data with RDF data descriptions. In: Proceedings of the 4th COLD Workshop (2013)
21. Schmidt, M., Meier, M., Lausen, G.: Foundations of SPARQL query optimization. In: Proceedings of the 13th ICDT, pp. 4–33. ACM (2010)
22. Seaborne, A.: SPARQL 1.1 Property Paths (2010). http://www.w3.org/TR/sparql11-property-paths/. Accessed Nov 2015
23. Soru, T., Marx, E., Ngomo, A.N.: ROCKER: a refinement operator for key discovery. In: Proceedings of the 24th WWW, pp. 1025–1033 (2015)
24. Soru, T., Marx, E., Ngonga Ngomo, A.-C.: Enhancing dataset quality using keys. In: Proceedings of the 14th ISWC, Posters & Demonstrations Track (2015)
25. Stickler, P.: CBD - Concise Bounded Description (2005). http://www.w3.org/Submission/CBD/. Accessed Oct 2015
26. Strong, D.M., Lee, Y.W., Wang, R.Y.: Data quality in context. Commun. ACM **40**(5), 103–110 (1997)
27. Symeonidou, D., Armant, V., Pernelle, N., Saïs, F.: SAKey: scalable almost key discovery in RDF data. In: Mika, P., et al. (eds.) ISWC 2014, Part I. LNCS, vol. 8796, pp. 33–49. Springer, Heidelberg (2014)

28. Tao, J., Sirin, E., Bao, J., McGuinness, D.L.: Extending OWL with integrity constraints. In: Description Logics, CEUR Workshop Proceedings, vol. 573. CEUR-WS.org (2010)
29. Thalheim, B.: Fundamentals of cardinality constraints. In: Pernul, G., Tjoa, A.M. (eds.) ER 1992. LNCS, vol. 645, pp. 7–23. Springer, Heidelberg (1992)
30. Töpper, G., Knuth, M., Sack, H.: DBpedia ontology enrichment for inconsistency detection. In: Proceedings of the 8th International Conference on Semantic Systems, I-SEMANTICS 2012, pp. 33–40. ACM, New York (2012)
31. Völker, J., Fleischhacker, D., Stuckenschmidt, H.: Automatic acquisition of class disjointness. Web Semant. Sci. Serv. Agents World Wide Web **35**(Part 2), 124–139 (2015). Machine Learning and Data Mining for the Semantic Web (MLDMSW)
32. Wang, R.Y., Strong, D.M.: Beyond accuracy: what data quality means to data consumers. J. Manage. Inf. Syst. **12**(4), 5–33 (1996)