Aspects of RDF Data Constraints in the Social, Behavioural and Economic Sciences

Thomas Bosch[1], Benjamin Zapilko[1], Joachim Wackerow[1], and Kai Eckert[2]

[1] GESIS – Leibniz Institute for the Social Sciences, Germany
{firstname.lastname}@gesis.org,
[2] Stuttgart Media University, Germany
eckert@hdm-stuttgart.de

Abstract. For research institutes, data libraries, and data archives, RDF data validation according to predefined constraints is a much sought-after feature, particularly as this is taken for granted in the XML world. Based on our work in the DCMI RDF Application Profiles Task Group and in cooperation with the W3C Data Shapes Working Group, we identified and published by today 82 types of constraints that are required by various stakeholders for data applications. In this paper, we formulate 246 constraints of 53 different types on six different vocabularies (Disco, QB, SKOS, PHDD, DCAT, XKOS) and classify them according to the complexity level of their type and their severity level. For 114 of these constraints, we evaluate the data quality of 15,694 data sets (4.26 billion triples) of research data for the social, behavioural, and economic (SBE) sciences obtained from 33 SPARQL endpoints. Based on the results, we formulate several hypotheses to direct the further development of constraint languages.

Keywords: Constraint Validation, Data Quality, RDF

1 Introduction

The social, behavioural, and economic sciences (SBE) require high-quality data for their empirical research. For more than a decade, members of the *SBE* community have been developing and using a metadata standard, composed of almost twelve hundred metadata fields, known as the *Data Documentation Initiative (DDI)*, an XML format to disseminate, manage, and reuse data collected and archived for research [11].

In XML, the definition of schemas containing data constraints and the validation of data according to these constraints is commonly used to ensure a certain level of data quality.

With the rise of the Web of Data, data professionals and institutions are very interested in having their data be discovered and used by publishing their data directly in RDF or at least publish accurate metadata about their data to facilitate data integration. Therefore, not only existing vocabularies like SKOS are used; recently, members of the *SBE* and Linked Data community developed with the *DDI-RDF Discovery Vocabulary (Disco)*[3] a means to expose *DDI* metadata as Linked Data.

For constraint formulation and validation on RDF data, several languages exist, like *Shape Expressions*, *Resource Shapes* or *Description Set Profiles*. *OWL 2* is also used as a constraint language under a closed world assumption. With its direct support of validation via SPARQL, *SPIN* is very popular and certainly plays an important role for future developments in this field.

In 2013, the W3C organized the RDF Validation Workshop[4], where experts from industry, government, and academia discussed first use cases for RDF constraint formulation and RDF data validation. In 2014, two working groups on RDF validation have been established to develop a language to express constraints on RDF data: the W3C RDF Data Shapes working group[5] and the DCMI RDF Application Profiles task group[6].

Bosch and Eckert [1] collected the findings of these working groups and initiated a database of RDF validation requirements which is available for contribution at http://purl.org/net/rdf-validation. The intention is to collaboratively collect case studies, use cases, requirements, and solutions regarding RDF validation in a comprehensive and structured way. The requirements are classified to better evaluate existing solutions and each requirement is directly mapped to a constraint type which is expressible by at least one constraint language. As *SPIN* is the within the RDF validation community accepted and widely adopted way to check constraints on RDF data, Bosch and Eckert [2] use *SPIN* as basis to define a validation environment (available at http://purl.org/net/rdfval-demo) in which the validation of any constraint language[7] can be implemented by representing them in SPARQL. The SPIN engine checks for each resource if it satisfies all constraints, which are associated with its assigned classes, and generates a result RDF graph containing information about all constraint violations.

The **contribution** of this paper is the development of a system

1. to classify RDF constraints (according to their severity levels) to evaluate the quality of metadata and data which may be represented by any vocabulary and
2. to classify RDF constraint types (according to their complexity) which in most cases correspond to RDF validation requirements[8] (sections 5 - 8).

By defining a huge amount of constraints of the majority of the constraint types, we apply the developed classification system to several and different vocabularies from the *SBE* domain to represent both metadata and data and therefore prove its generality. A complex and complete real world running example from the

[3] http://rdf-vocabulary.ddialliance.org/discovery.html
[4] http://www.w3.org/2012/12/rdf-val/
[5] http://www.w3.org/2014/rds/charter
[6] http://wiki.dublincore.org/index.php/RDF-Application-Profiles
[7] The only limitation is that constraint languages must be represented in RDF
[8] For simplicity reasons, we use the terms *constraint types* and *constraints* instead of *RDF constraint types* and *RDF constraints* in the rest of the paper

SBE domain serves to prove the claim that the developed classification system perfectly applies for diverse vocabularies. We describe why RDF validation is important for the *SBE* community (section 2), how data in tabular format and metadata on person-level data sets, aggregated data sets, and thesauri are represented in RDF, how therefore reused vocabularies are interrelated (section 4), and how *SBE* (meta)data is validated against constraints of different constraint types (sections 5 - 8). We evaluated the (meta)data quality of large real world data sets (more than 4.2 billion triples and 15 thousand data sets) from the *SBE* domain represented by multiple vocabularies to get an understanding (1) which sets of constraint types (on different levels of complexity) and (2) which sets of constraints (associated with particular severity levels) encompass the constraints causing the most/fewest constraint violations (see section 10).

In this paper, we discuss constraints on RDF data in general. Note that the data represented in RDF can be data in the sense of SBE sciences, but also metadata about published or unpublished data. We generally refer to both simply as RDF data and only distinguish between data and metadata in the data set descriptions and in the case that it matters for the purpose of this paper.

2 Motivation

The data most often used in research within the *SBE* community is *person-level data* (or more generally *record-unit data*, i.e., data collected about individuals, businesses, and households) in form of responses to studies or taken from administrative registers (such as hospital records, registers of births and deaths). The range of person-level data is very broad - including census, education, health data and business, social, and labor force surveys. This type of research data is held within data archives or data libraries after it has been collected, so that it may be reused by future researchers.

By its nature, person-level data is highly confidential and access is often only permitted for qualified researchers who must apply for access. Researchers typically represent their results as aggregated data in form of multi-dimensional tables with only a few columns; so-called *variables* such as *sex* or *age*. Aggregated data, which answers particular research questions, is derived from person-level data by statistics on groups or aggregates such as frequencies and arithmetic means. The purpose of publicly available aggregated data is to get a first overview and to gain an interest in further analyses on the underlying person-level data. Aggregated data is more and more published in form of CSV files, allowing to perform data calculations.Portals harvest metadata (as well as publicly available data) from multiple RDF data providers. To ensure high quality, (meta)data must satisfy certain criteria - specified in terms of RDF constraints.

For more detailed analyses, researchers refer to person-level data including additional variables needed to answer subsequent research questions like the comparison of studies between countries. A *study* represents the process by which a data set was generated or collected. Eurostat[9], the statistical office of the Eu-

Kai: abgeshen davon dass ich more and more nicht mehr sehen kann, ist der zusammenhang mit RDF daten und damit RDF providern im nächsten satz unklar

Kai: verwirrend. wie gesagt, können wir nicht bei RDF data bleiben? was anderes behandelt ihr doch eh nicht und qb enthält doch keine metadaten.

[9] http://ec.europa.eu/eurostat

ropean Union, provides research findings in form of aggregated data (downloadable as CSV files) and its metadata at European level that enable comparisons between countries. The variable *formal childcare*[10] captures the measured availability of childcare services in percent over the population in European Union member states by the variables *year*, *duration* (in hours per week), *age* of the child, and *country*. Variables are constructed out of values (of one or multiple datatypes) and/or code lists. The variable *age*, e.g., may be represented by values of the datatype *xsd:nonNegativeInteger* or by a code list of age clusters (e.g., '0 to 10' and '11 to 20').

To determine if variables measuring *age* - collected for different countries (age_{DE}, age_{UK}) - are comparable, diverse constraints are checked: (1) variable definitions must be available, (2) for each code a human-readable label has to be specified, (3) code lists must be structured properly, and (4) code lists must either be identical or at least similar. If a researcher only wants to get a first overview over comparable variables (use case 1), covering the first three constraints may be sufficient, i.e., the violation of the first three constraints is more serious than the violation of the last constraint. If the intention of the researcher is to perform more sophisticated comparisons (use case 2), however, the user may raise the severity level of the last constraint.

Kai: ich finde das immer noch recht verwirrend. Ist das mit den aggregierten Daten wichtig für dieses Paper? Wieso werden constraints nur gechecked wenn variablen auf vergleichbarkeit geprüft werden? das habt ihr doch gar nicht gemacht sondern ganz allgemein constraints für vokabulare aufgestellt. Ist QB und Disco nicht sowieso unabhängig von konkreten Variablen?

3 Related Work

For data archives, research institutes, and data libraries, RDF validation according to predefined constraints is a much sought-after feature, particularly as this is taken for granted in the XML world. DDI-XML documents, e.g., are validated against diverse XSDs[11]. As certain constraints cannot be formulated and validated by XSDs, so-called secondary-level validation tools like *Schematron*[12] have been introduced to overcome the limitations of XML validation. *Schematron* generates validation rules and validates XML documents according to them. With RDF validation, one can overcome drawbacks when validating XML documents[13]. It cannot be validated, e.g., if each code of a variable's code list is associated with a category (*R-86*). Additionally, it cannot be validated that if an element has a specific value, then certain child elements must be present (*R-71*). A comprehensive comparison of XML and RDF validation, however, is not within the scope of this paper.

A well-formed *RDF Data Cube* is an a RDF graph describing one or more instances of *qb:DataSet* for which each of the 22 integrity constraints[14], defined

[10] Aggregated data and its metadata is available at: http://ec.europa.eu/eurostat/web/products-datasets/-/ilc_caindformal

[11] http://www.ddialliance.org/Specification/

[12] https://msdn.microsoft.com/en-us/library/aa468554.aspx

[13] http://www.xmlmind.com/xmleditor/_distrib/doc/xmltool/xsd_structure_limitations.html

[14] http://www.w3.org/TR/vocab-data-cube/#wf

within the *QB* specification, passes. Each integrity constraint is expressed as narrative prose and, where possible, a SPARQL ASK query or query template. If the ASK query is applied to an RDF graph then it will return true if that graph contains one or more *QB* instances which violate the corresponding constraint [7]. Mader, Haslhofer, and Isaac investigated how to support taxonomists in improving SKOS vocabularies by pointing out quality issues that go beyond the integrity constraints defined in the SKOS specification [9].

4 Common Vocabularies in SBE Sciences

Thomas: Ich habe XKOS und DCAT noch ganz am Ende eingeführt (jeweils nur 1 Satz)

For our evaluation, we examined six different vocabularies commonly used in or developed for the SBE sciences which are briefly introduced in the following. For three of them, we analysed actual data according to constraint violations. For PHDD, there is not yet enough data available so we limit our examination to the vocabulary itself.

The *RDF Data Cube Vocabulary (QB)*[15] is a W3C recommendation for representing metadata on *data cubes*, i.e. multi-dimensional aggregated data, in RDF [6]. A *qb:DataStructureDefinition* contains metadata of the data collection. The variable *formal childcare* is modelled as *qb:measure*, since it stands for what has been measured in the data collection. The variables *year*, *duration*, *age*, and *country* are *qb:dimensions*. Data values, i.e., the availability of childcare services in percent over the population, are collected in a *qb:DataSet*. Each data value is represented inside a *qb:Observation* which contains values for each dimension[16].

Kai: Wieso springt das von einer sehr knappen Einführung direkt zu konkreten (beliebigen?) Variablen? Ich möchte wissen, wofür QB benutzt wird und gerne ein konkretes Beispiel sehen, um mir einen Data Cube vorstellen zu können.

Physical Data Description (PHDD)[17] is a vocabulary to represent data in tabular format in RDF enabling further aggregations and calculations. The data could be either represented in records with character-separated values (CSV) or fixed length. *Eurostat* provides a CSV file, a two-dimensional table (*phdd:Table*) about *formal childcare* which is structured by a table structure (*phdd:TableStructure*, *phdd:Delimited*) including information about the character set (*ASCII*), the variable delimiter (*,*), the new line marker (*CRLF*), and the first line where the data starts (*2*). The table structure is related to table columns (*phdd:Column*) which are described by column descriptions (*phdd:DelimitedColumnDescription*). For the column containing the cell values in percent, the column position (*5*), the recommended data type (*xsd:nonNegativeInteger*), and the storage format (*TINYINT*) is stated.

For more detailed analyses we refer to the metadata on person-level data collected for the series *EU-SILC (European Union Statistics on Income and Living Conditions)*[18]. Where data collection is cyclic, data sets may be released as *se-*

[15] http://www.w3.org/TR/vocab-data-cube/
[16] The complete running example in RDF is available at: https://github.com/boschthomas/rdf-validation/tree/master/data/running-example
[17] https://github.com/linked-statistics/physical-data-description
[18] http://www.gesis.org/missy/eu/metadata/EU-SILC

Kai: Wieso jetzt wieder qb?

Thomas: ich möchte zeigen wie aggregierte daten und personenbezogene daten in RDF verbunden sind. Weglassen?

ries, where each cycle produces one or more data sets. Aggregated (qb:DataSet) and underlying person-level data sets (*disco:LogicalDataSet*) are connected by *prov:wasDerivedFrom*. The aggregated variable *formal childcare* is calculated on the basis of six person-level variables (e.g., *Education at pre-school*[19]) for which detailed metadata is given (e.g., code lists) enabling researchers to replicate the results shown in aggregated data tables. The *DDI-RDF Discovery Vocabulary (Disco)* is a vocabulary to represent metadata on person-level data in RDF. The series (*disco:StudyGroup*) *EU-SILC* contains one study (*disco:Study*) for each year (*dcterms:temporal*) of data collection. *dcterms:spatial* points to the countries for which the data has been collected. The study *EU-SILC 2011* contains eight person-level data sets (*disco:LogicalDataSet*) including person-level variables (*disco:Variable*) like the six ones needed to calculate the aggregated variable *formal childcare*.

The *Simple Knowledge Organization System (SKOS)* is reused multiple times to build *SBE* vocabularies. The codes of the variable *Education at pre-school* (number of education hours per week) are modeled as *skos:Concepts* and a *skos:OrderedCollection* organizes them in a particular order within a *skos:memberList*. A variable may be associated with a theoretical concept (*skos:Concept*). *skos:narrower* builds the hierarchy of theoretical concepts within a *skos:ConceptScheme* of a series. The variable *Education at pre-school* is assigned to the theoretical concept *Child Care* which is the narrower concept of *Education* - one of the top concepts of the series *EU-SILC*. Controlled vocabularies (*skos:ConceptScheme*), serving as extension and reuse mechanism, organize types (*skos:Concept*) of descriptive statistics (*disco:SummaryStatistics*) like minimum, maximum, and arithmetic mean. *XKOS*[20] is a SKOS extension to describe formal statistical classifications like the International Standard Classification of Occupations (*ISCO*). and the Statistical Classification of Economic Activities in the European Community *NACE*. *DCAT* enables to represent data sets inside of data collections like portals, repositories, catalogs, and archives which serve as typical entry points when searching for data.

5 Classification of Constraint Types and Constraints

Bosch et al. identified 76 requirements to formulate RDF constraints (e.g. *R-75: minimum qualified cardinality restrictions*); each of them corresponding to an RDF constraint type[21] [3]. We published a technical report[22] in which we explain each requirement/constraint type in detail and give examples for each expressed by different constraint languages. The knowledge representation formalism *Description logics (DL)*, with its well-studied theoretical properties, provides the

[19] http://www.gesis.org/missy/eu/metadata/EU-SILC/2011/Cross-sectional/original#2011-Cross-sectional-RL010

[20] https://github.com/linked-statistics/xkos

[21] Constraint types and constraints are uniquely identified by alphanumeric technical identifiers like *R-71-CONDITIONAL-PROPERTIES*

[22] Available at: http://arxiv.org/abs/1501.03933

foundational basis for each constraint type. Therefore, this technical report contains mappings to *DL* to logically underpin each requirement and to determine which *DL* constructs are needed to express each constraint type [3]. We classified both RDF constraint types and RDF constraints to gain better insights into the quality of RDF data with respect to this classification, independent of the used vocabulary. We recently published a technical report[23] (serving as first appendix of this paper) in which we describe 246 constraints of 53 distinct constraint types on six vocabularies [5].

5.1 Classification of RDF Constraint Types

According to the complexity of constraint types, the complete set of *constraint types* encompasses three disjoint *sets of constraint types*:

1. ***Vocabulary Constraint Types***
2. ***Simple Constraint Types***
3. ***Complex Constraint Types***

The modeling languages *RDF*, *RDFS*, and *OWL* are typically used to define vocabularies. *Vocabulary constraint types* denotes the set of constraint types whose constraints can be extracted completely automatically out of the formal specifications of vocabularies. As vocabularies have been specified using *RDF*, *RDFS*, and *OWL*, *vocabulary constraints* ensure that the data is consistent with the intended syntaxes, semantics, and integrity of vocabularies' data models. *Minimum qualified cardinality restrictions* (*R-74*), e.g., guarantee that individuals of given classes are connected by particular properties to at least n different individuals/literals of certain classes or data ranges. This way, it is expressible in *OWL 2* that a *phdd:TableStructure* has (*phdd:column*) at least one *phdd:Column*:

Kai: Ich dachte, das kommt aus dem Vokabular. Dann muss das heißen: For example, in PHDD, a minimum qualified cardinality constraint can be obtained from the OWL definition of this restriction class, ensuring that a *phdd:TableStructure* has (*phdd:column*) at least one *phdd:Column*:

```
[   a owl:Restriction ; rdfs:subClassOf TableStructure ;
    owl:minQualifiedCardinality 1 ;
    owl:onProperty column ;
    owl:onClass Column ] .
```

Simple and *complex constraints* are in contrast to *vocabulary constraints* not explicitly defined within formal specifications of vocabularies. *Simple* and *complex constraints* are defined according to textual descriptions of the intended semantics of vocabularies. *Simple constraint types* is the set of constraint types whose constraints can be easily defined without much effort in addition to *vocabulary constraints*. *Data property facets* (*R-46*) is an example of a *simple constraint type* which enables to declare frequently needed facets for data properties in order to validate input against simple conditions including min/max values, regular expressions, and string length. The abstract of series/studies, e.g., should have a minimum length.

Kai: einfügen: the already defined (auf den Lesefluss achten, das ist eine sehr pragmatische klassifikation und muss nicht übertrieben formal und trocken geschrieben sein)

Complex constraint types encompass constraint types for which the definition of constraints is rather complex and cannot be derived from vocabulary

[23] Available at: http://arxiv.org/abs/1504.04479

definitions. For assessing the quality of thesauri, e.g., we concentrate on the graph-based structure and apply graph- and network-analysis techniques. An example of such constraints of the constraint type *structure* is that a thesaurus should not contain many orphan concepts, i.e., concepts without any associative or hierarchical relations, lacking context information valuable for search. *Complex constraints* show the importance to develop constraint languages enabling to describe more complex constraints.

Kai: der satz macht so keinen sinn... aber er ist ein weiteres gutes beispiel, warum du viel mehr auf die formulierungen achten musst. Ich weiß genau was du meinst, du hast auch recht, aber hier steht trotzdem ein sinnloser satz, so wie er formuliert ist.

5.2 Classification of RDF Constraints

Kai: ich würde da auch für simple und vor allem für complex konkrete beispiele in code einfügen

SBE experts determined the default ***severity level*** (*R-158*) for each of the 246 constraints to indicate how serious the violation of the constraint is. We propose an extensible metric to measure the continuum of severity levels ranging from *informational* to *error*.According to the default severity level of constraints, the complete set of constraints encompasses three disjoint *sets of constraints*:

Kai: da muss mindestens ein überleitender satz rein, der noch mal den unterschied von constraint types und types aufgreift und erklärt, dass es jetzt um konkrete constraints auf den vokabularen geht, deren severity eben vom konkreten kontext abhängt und nicht allgemein über den type bestimmt werden kann.

Kai: oder etwas weniger anmaßend: we use the commonly accepted classification of log messages in software development and distingush informational, warning and error.

- ***informational constraints***: constraints with severity level *informational*
- ***warning constraints***: constraints with severity level *warning*
- ***error constraints***: constraints with severity level *error*

Violations of *informational constraints* point to desirable but not necessary data improvements to achieve RDF representations which are ideal in terms of syntax and semantics of used vocabularies. Data not conforming to *warning* and *error constraints* is syntactically and/or semantically not correctly represented. Data not conforming to *warning constraints* but to *error constraints* could be processed further. Data not corresponding to *error constraints*, however, cannot be processed further after validation. As the purpose of *vocabulary constraints* is to ensure explicitly stated semantics of vocabularies, their default severity levels are in most cases very strong (*error*) and in average stronger than the severity levels of *simple* and *complex constraints.* As a consequence, violating many *vocabulary constraints* is an indicator for bad (meta)data quality[24].

Kai: auch dieser satz macht keinen sinn für mich, ebensowenig wie die

Although, we provide default severity levels for each constraint, validation environments should enable users to adapt the severity levels of constraints according to their individual needs. Validation environments should enable users to select which constraints to validate against depending on their individual use cases. For some use cases, validating *vocabulary constraints* may be more important than validating *simple* or *complex constraints.* For other use cases, validating *error constraints* may be sufficient without taking *warning* and *informational constraints* into account. We evaluated the (meta)data quality of large real world data sets represented by multiple and different vocabularies to get an understanding (1) which sets of constraint types (on different levels of

[24] For simplicity reasons, we only assign severity levels to *vocabulary constraints* in this paper in case they differ from *error constraints.*

complexity) and (2) which sets of constraints (associated with particular severity levels) encompass the constraints causing the most/fewest constraint violations (see section 10).

6 Vocabulary Constraint Types

The constraint type *vocabulary* guarantees that users do not invent new or use deprecated terms of vocabularies. *Value is valid for datatype* (*R-223*) constraints serve to make sure that all literal values are valid with regard to their datatypes - as stated in the vocabularies. Thus, it is checked that all date values (e.g., *dcterms:date*) are actually of the datatype *xsd:date* and that *xsd:nonNegativeInteger* values (e.g. *disco:frequency*) are not negative. Depending on property datatypes, two different literal values have a specific ordering with respect to operators like $<$(*R-43: literal value comparison*). Start dates (*disco:startDate*), e.g., must be before ($<$) end dates (*disco:endDate*).

$$\begin{aligned}\mathcal{K} = \{\ & \texttt{isStructuredBy} \sqsubseteq \neg\ \texttt{column},\\ & \texttt{TableDescription} \sqcap \texttt{ColumnDescription} \sqsubseteq \bot,\\ & \texttt{CategoryStatistics} \equiv\\ & \forall\ \texttt{computationBase}.\{\texttt{valid},\texttt{invalid}\} \sqcap \texttt{langString}\ \}\end{aligned}$$

All properties, not having the same domain and range types, are defined to be pairwise disjoint (*R-9: disjoint properties*), i.e., no individual x can be connected to an individual/literal y by disjoint properties (e.g., *phdd:isStructuredBy* and *phdd:column*). All *PHDD* classes (e.g., *phdd:TableDescription*, *phdd:ColumnDescription*) are pairwise disjoint (*R-7: disjoint classes*), i.e., individuals cannot be instances of multiple disjoint classes. It is a common requirement to narrow down the value space of properties by an exhaustive enumeration of valid values (*R-30/37: allowed values*). *disco:CategoryStatistics*, e.g., can only have *disco:computationBase* relationships to the values *valid* and *invalid* of the datatype *rdf:langString*. Validation should *exploit sub-super relations* in vocabularies (*R-224*). If *dcterms:coverage* and one of its sub properties (*dcterms:spatial*, *dcterms:temporal*) are given, it is checked that *dcterms:coverage* is not redundant with its sub-properties which may indicate when the data is verbose/redundant or expressed at a too general level.

Thomas: K einführen

—

Property domain (*R-25, R-26*) and *range* (*R-28, R-35*) constraints restrict domains and ranges of properties. Only *phdd:Tables*, e.g., can have *phdd:isStructuredBy* relationships and *xkos:belongsTo* relationships can only point to *skos:Concepts*:

$$\exists\ \text{isStructuredBy}.\top \sqsubseteq \text{Table}$$
$$\top \sqsubseteq \forall\ \text{belongsTo}.\text{Concept}$$

A *universal quantification* (*R-91*) contains all those individuals that are connected by a property only to individuals/literals of particular classes or data ranges. Only *dcat:Catalogs*, e.g., can have *dcat:dataset* relationships to *dcat:Datasets*:

$$\text{Catalog} \sqsubseteq \forall \text{ dataset.Dataset}$$

Existential quantifications (*R-86*) enforce that instances of given classes must have some property relation to individuals/literals of certain types. Variables, e.g., should have a relation to a theoretical concept ($\mathcal{SL}_0$). The variable *Education at pre-school* is associated with the theoretical concept *Child Care*. The default severity level of the constraint is weak, as in most cases research can be continued without having information about the theoretical concept of a variable.

$$\text{Variable} \equiv \exists \text{ concept.Concept}$$

Minimum/maximum/exact qualified cardinality restrictions (*R-74, R-75, R-76*) contain all those individuals that are connected by a property to at least/at most/exactly n different individuals/literals of particular classes or data ranges. A *phdd:TableStructure* has (*phdd:column*) at least one *phdd:Column*, a *disco:Variable* has at most one *disco:concept* relationship to a theoretical concept (*skos:Concept*), and a *qb:DataSet* is structured by (*qb:structure*) exactly one *qb:DataStructureDefinition*.

$$\text{TableStructure} \sqsubseteq \geqslant 1 \text{ column.Column}$$
$$\text{Variable} \sqsubseteq \leqslant 1 \text{ concept.Concept}$$
$$\text{DataSet} \sqsubseteq \geqslant 1 \text{ structure.DSD} \sqcap \leqslant 1 \text{ structure.DSD}$$

Some constraint types enable performing reasoning prior to validation which may resolve or cause constraint violations. With *subsumption* (*R-100*), one can state that *xkos:ClassificationLevel* is a sub-class of *skos:Collection*, i.e., each *xkos:ClassificationLevel* must also be part of the *skos:Collection* class extension. With *sub properties* (*R-54, R-64*), one can state that *disco:fundedBy* is a sub-property of *dcterms:contributor* - i.e., if a study is funded by an organization, then this organization contributed to this study.

$$\text{ClassificationLevel} \sqsubseteq \text{Collection}$$
$$\text{fundedBy} \sqsubseteq \text{contributor}$$

7 Simple Constraint Types

$\mathcal{CT}_S$ is the set of constraint types whose constraints can be easily defined without much effort in addition to $\mathcal{CT}_B$ constraints. For data properties, it may be desirable to restrict that values of predefined languages must be present for determined number of times (*R-48/49: language tag cardinality*): (1) It is checked if literal language tags are set. Some controlled vocabularies, e.g., contain literals in natural language, but without information what language has actually been used ($\mathcal{SL}_1$). (2) Language tags must conform to language standards ($\mathcal{SL}_2$). (3) Some thesaurus concepts are labeled in only one, others in multiple languages. It may be desirable to have each concept labeled in each of the languages that are also used on the other concepts, as language coverage incompleteness for some concepts may indicate shortcomings of thesauri ($\mathcal{SL}_0$) [9].

Default values (*R-31, R-38*) for objects/literals of given properties are inferred automatically when properties are not present in the data. The value *true* for the property *disco:isPublic* indicates that a *disco:LogicalDataSet* can be accessed by anyone. Per default, however, access to data sets should be restricted (*false*) ($\mathcal{SL}_0$). Many properties are not necessarily required but *recommended* within a particular context (*R-72*). The property *skos:notation*, e.g., is not mandatory for *disco:Variable*s, but recommended to represent variable names ($\mathcal{SL}_0$). Percentage values are only valid when they are within the literal range of 0 and 100 (*R-45: literal ranges*; $\mathcal{SL}_2$) which is checked for *disco:percentage* standing for the number of cases of a given code in relation to the total number of cases for a particular variable.

$$\mathcal{K} = \{\ \texttt{(funct identifier}^{-}\texttt{), identifier keyfor Resource}\ \}$$

It is often useful to declare a given (data) property as the *primary key* (*R-226*) of a class, so that a system can enforce uniqueness and build URIs from user inputs and imported data. In *Disco*, resources are uniquely identified by the property *adms:identifier*, which is therefore inverse-functional, i.e., for each *rdfs:Resource* x, there can be at most one distinct resource y such that y is connected by *adms:identifier*$^{-}$ to x ($\mathcal{SL}_2$). Keys, however, are even more general than *inverse-functional properties* (*R-58*), as a key can be a data property, an object property, or a chain of properties [10]. Thus and as there are different sorts of key, and as keys can lead to undecidability, *DL* is extended with the construct *keyfor* [8] which is implemented by the *OWL 2 hasKey* construct.

8 Complex Constraint Types

$\mathcal{CT}_C$ denotes the set of constraint types for which the definition of constraints is rather complex and cannot be derived from vocabulary definitions. *Data model consistency* constraints ensure the integrity of the data according to the intended semantics of vocabularies. Every *qb:Observation*, e.g., must have a value for each dimension declared in its *qb:DataStructureDefinition* ($\mathcal{SL}_2$) and no two *qb:Observations* in the same *qb:DataSet* can have the same value for all dimensions ($\mathcal{SL}_1$). If a *qb:DataSet D* has a *qb:Slice S*, and S has an *qb:Observation O*, then the *qb:DataSet* corresponding to O must be D ($\mathcal{SL}_1$). *Mathematical Operations* (*R-41, R-42*; e.g. date calculations and statistical computations like average, mean, and sum) are performed to ensure the integrity of data models. The sum of percentage values of all variable codes, e.g., must exactly be 100 ($\mathcal{SL}_2$) and the minimum absolute frequency of all variable codes do not have to be greater than the maximum ($\mathcal{SL}_2$).

In many cases, resources must be *members of controlled vocabularies* (*R-32*). If a dimension property, e.g., has a *qb:codeList*, then the value of the dimension property on every *qb:Observation* must be in the code list ($\mathcal{SL}_2$). Summary statistics types like minimum, maximum, and arithmetic mean are maintained within a controlled vocabulary. Thus, summary statistics can only have *disco:summaryStatisticType* relationships to *skos:Concept*s which must be members of the controlled vocabulary *ddicv:SummaryStatisticType*, a *skos:ConceptScheme* ($\mathcal{SL}_2$). Objects/literals can be declared to be ordered for given properties (*R-121/217: ordering*). Variables, questions, and codes, e.g., are typically organized in a particular order. If codes (*skos:Concept*) should be ordered, they must be members (*skos:memberList*) in an ordered collection (*skos:OrderedCollection*), the variable's code list ($\mathcal{SL}_0$).

It is useful to declare properties to be *conditional* (*R-71*), i.e., if particular properties exist (or do not exist), then other properties must also be present (or absent). To get an overview over a series/study either an abstract, a title, an alternative title, or links to external descriptions should be provided. If an abstract and an external description are absent, however, a title or an alternative title should be given ($\mathcal{SL}_1$). In case a variable is represented in form of a code list, codes may be associated with categories, i.e., human-readable labels ($\mathcal{SL}_0$). The variable *Education at pre-school*, e.g., is represented as ordered code list without any categories. If a *skos:Concept* represents a code (having *skos:notation* and *skos:prefLabel* properties), then the property *disco:isValid* has to be stated indicating if the code stands for valid (*true*) or missing (*false*) cases ($\mathcal{SL}_2$). *Context-specific exclusive or of property groups* (*R-11*) constraints restrict individuals of given classes to have properties defined within exactly one of multiple property groups. *skos:Concept*s can have either *skos:definition* (when interpreted as theoretical concepts) or *skos:notation* and *skos:prefLabel* properties (when interpreted as codes/categories), but not both ($\mathcal{SL}_2$).

9 Implementation

SPARQL is generally seen as the method of choice to validate RDF data according to certain constraints. We use *SPIN*, a SPARQL-based way to formulate and check constraints, as basis to develop a validation environment (available at http://purl.org/net/rdfval-demo)[25] to validate RDF data according to constraints expressed my arbitrary constraint languages like Shape Expressions, Resource Shapes, and the Web Ontology Language[26] [2]. The *RDF Validator* also validates RDF data to ensure correct syntax, semantics, and integrity of diverse vocabularies such as *Disco*, *QB*, *SKOS*, and *PHDD*. Although accessible within our validation tool, we provide all implemented constraints[27] in form of SPARQL CONSTRUCT queries. For the subsequent evaluation, we implemented 213 constraints on *Disco*, *QB*, *SKOS*, and *PHDD* data sets. The SPIN engine checks for each resource if it satisfies all constraints, which are associated

[25] Source code downloadable at: https://github.com/boschthomas/rdf-validator
[26] SPIN mappings available at: https://github.com/boschthomas/rdf-validation/tree/master/SPIN
[27] https://github.com/boschthomas/rdf-validation/tree/master/constraints

with its assigned classes, and generates a result RDF graph containing information about all constraint violations. There is one SPIN construct template for each constraint type and vocabulary-specific constraint[28]. A SPIN construct template contains a SPARQL CONSTRUCT query which generates constraint violation triples indicating the subject and the properties causing constraint violations, and the reason why constraint violations have been raised. A SPIN construct template creates constraint violation triples if all triple patterns within the SPARQL WHERE clause match. *Missy*[29] provides comprehensive Linked Data services like diverse RDF exports of person-level metadata conforming to the *Disco* vocabulary in form of multiple concrete syntaxes.

10 Evaluation

10.1 Evaluation Setup

In close collaboration with several *SBE* domain experts, we defined 246 constraint of 53 different types on six vocabularies (*Disco*, *QB*, *SKOS*, *PHDD*, *DCAT*, *XKOS*) and classified them according to the complexity level of their type and their severity level. For 114 of these constraints, we evaluated the data quality of 15,694 data sets (4.26 billion triples) of *SBE* research data on three common vocabularies in *SBE* sciences (*Disco*, *QB*, *SKOS*) obtained from 33 SPARQL endpoints. We distinct two classes of vocabularies: (1) well-established vocabularies (e.g., *QB*, *SKOS*) which are widely adopted and accepted and (2) newly developed vocabularies (e.g., *Disco* which will be published in 2015) which are either recently published or are still in the publication process.

We validated 9,990 / 3,775,983,610 (*QB*), 4,178 / 477,737,281 (*SKOS*), and 1,526 / 9,673,055 (*Disco*) data sets / triples using the *RDF Validator* in batch mode. We validated, i.a., (1) *QB* data sets published by the *Australian Bureau of Statistics (ABS)*, the *European Central Bank (ECB)*, and the *Organisation for Economic Co-operation and Development (OECD)*, (2) *SKOS* thesauri like the *AGROVOC Multilingual agricultural thesaurus*, the *STW Thesaurus for Economics*, and the *Thesaurus for the Social Sciences (TheSoz)*, and (3) *Disco* data sets provided by the *Microdata Information System (Missy)*, the *DwB Discovery Portal*, the *Danish Data Archive (DDA)*, and the *Swedish National Data Service (SND)*. We recently published a technical report[30] (serving as second appendix of this paper) in which we describe the evaluation in detail [4]. As we evaluated nearly 10 thousand *QB* data sets, we published the evaluation results for each data set in form of one document per SPARQL endpoint[31].

[28] For a comprehensive description of the *RDF Validator*, we refer to [2]
[29] http://www.gesis.org/missy/eu/missy-home
[30] Available at: http://arxiv.org/abs/1504.04478
[31] Available at: https://github.com/boschthomas/rdf-validation/tree/master/evaluation/data-sets/data-cube

10.2 Evaluation Results and Formulation of Hypotheses

Table 2 shows the results of the evaluation, more specifically the constraints and the constraint violations, which are caused by these constraints, in percent. The constraints and their raised constraint violations are grouped by vocabulary, complexity level of their type, and their severity level. The number of evaluated triples and data sets differs between the vocabularies as we evaluated 3.8 billion *QB*, 480 million *SKOS*, and 10 million *Disco* triples. To be able to formulate hypotheses which apply for all vocabularies, we only use normalized relative values representing the percentage of constraints and constraint violations belonging to the respective class

	Disco		***QB***		***SKOS***		***Total***	
	C	CV	C	CV	C	CV	C	CV
	143	3,575,002	35	45,635,861	35	5,540,988	213	54,751,851
complex	25.9%	18.3%	37.1%	**100%**	**37.1%**	21.4%	33.4%	**46.6%**
simple	19.6%	15.7%	8.6%	0.0%	**34.3%**	**78.6%**	20.8%	31.4%
vocabulary	**54.6%**	**66.1%**	**54.3%**	0.0%	**28.6%**	0.0%	**45.8%**	22.0%
info	**52.5%**	**52.6%**	11.4%	0.0%	**60.0%**	41.2%	**41.3%**	31.3%
warning	7.0%	29.4%	8.6%	**99,8%**	14.3%	**58.8%**	10.0%	**62.7%**
error	40.6%	18%	**80.0%**	0.3%	25.7%	0.0%	**48.8%**	6.1%

C (constraints), CV (constraint violations)

Table 1: Constraints and Constraint Violations

As the evaluation is based on three vocabularies, we cannot make valid general statements for all vocabularies, but we can formulate several hypotheses to direct the further development of constraint languages. As these hypotheses cannot be proved yet, they still have to be verified or falsified by evaluating the quality of data represented by further well-established and newly developed vocabularies. Almost 1/2 of all 213 constraints and more than 50% of the *Disco* and the *QB* constraints are *vocabulary constraints*. The *SKOS* constraints are nearly to the same extend *vocabulary*, *simple*, and *complex constraints*.

Hypothesis 1 *As a significant amount of 46 % of the constraints are vocabulary constraints which can be expressed by modeling languages like RDF, RDFS, and OWL 2, the further development of constraint languages should concentrate on expressing simple and especially complex constraints which up to now in most cases can only be expressed by plain SPARQL.*

Nearly 1/2 of all violations are caused by *complex constraints*, 1/3 by *simple constraints*, and 1/5 by *vocabulary constraints*. The fact that only 1/5 of all violations result from *vocabulary constraints*, even though, 46% of all constraints are *vocabulary constraints*, indicates good data quality for all vocabularies with regard to their formal specifications.

Hypothesis 2 *For all vocabularies, data corresponds to their formal specifications which demonstrates that constraint formulation in general works.*

2/3 of the *Disco* violations result from *vocabulary constraints*, *QB* violations are almost only raised by *complex constraints*, and nearly 80% of the *SKOS* violations are caused by *simple constraints*. For well-established vocabularies, *vocabulary constraints* are almost completely satisfied[32] which indicates good data quality according to formal specifications of vocabularies. For newly defined vocabularies, however, 2/3 of all violations are raised by *vocabulary constraints* which indicates bad data quality with regard to the formal specifications of vocabularies.

Hypothesis 3 *Data represented by well-established vocabularies corresponds to formal specifications of these vocabularies which demonstrates that constraint formulation in general works. Data represented by newly developed vocabularies, in contrast, does not correspond to formal specifications of these vocabularies.*

It is likely that a newly developed vocabulary is still subject of constant change and that early adopters did not properly understand its formal specification. Thus, published data may not be consistent with its current draft or version. In case newly developed vocabularies turn into well-established ones, data providers are experienced in publishing their data in conformance with these vocabularies, and the formal specifications are more elaborated and therefore clearer. As a consequence, vocabulary constraints are satisfied which leads to better data quality.

Hypothesis 4 *A significant amount of 47% of the violations refer to complex constraints that are not easily, concisely, and intuitively expressible in existing constraint languages which confirms the necessity to provide suitable constraint languages.*

In general, *vocabulary* and *simple constraints* are easily, concisely, and intuitively expressible by either modeling languages (e.g., *RDF*, *RDFS*, *OWL 2*) or constraint languages (e.g., *ShEx*, *ReSh*, *SPIN*). This is not the case, however, for *complex constraints* which in most cases are still only expressible by plain *SPARQL*. As almost 1/2 of all violations are caused by *complex constraints*, data quality can be significantly improved when suitable constraint languages are developed which enable to define *complex constraints* in an easy, concise, and intuitive way.

[32] e.g. only 1.777 *QB* violations

	Disco		***QB***		***SKOS***		***Total***	
	C	CV	C	CV	C	CV	C	CV
	143	3,575,002	35	45,635,861	35	5,540,988	213	54,751,851
complex	25.9%	18.3%	37.1%	**100%**	**37.1%**	21.4%	33.4%	**46.6%**
simple	19.6%	15.7%	8.6%	0.0%	**34.3%**	**78.6%**	20.8%	31.4%
vocabulary	**54.6%**	**66.1%**	**54.3%**	0.0%	**28.6%**	0.0%	**45.8%**	22.0%
info	**52.5%**	**52.6%**	11.4%	0.0%	**60.0%**	41.2%	**41.3%**	31.3%
warning	7.0%	29.4%	8.6%	**99,8%**	14.3%	**58.8%**	10.0%	**62.7%**
error	40.6%	18%	**80.0%**	0.3%	25.7%	0.0%	**48.8%**	6.1%

C (constraints), CV (constraint violations)

Table 2: Constraints and Constraint Violations

1/2 of all constraints are *error constraints* and 10% of all constraints are *warning constraints*. *Informational constraints* caused 1/3 and *warning constraints* 2/3 of all violations. As the percentage of severe violations is very low for all vocabularies, data quality is high with regard to the severity level of constraints.

Hypothesis 5 *The percentage of severe violations is very low, compared to about 2/3 of warning violations and 1/3 of informational violations, which implies that proper constraint languages can significantly improve the data quality beyond fundamental requirements.*

80% of the *QB* constraints are error constraints. More than 50% of the *Disco* and *SKOS* constraints, however, are informational constraints. 1/6 of the Disco violations are caused by error constraints. Almost all *QB* violations and 59% of the *SKOS* violations are caused by warning constraints.

Hypothesis 6 *For well-established vocabularies, data quality is high as serious violations rarely appear. For newly developed vocabularies, data quality is worse as serious violations occur partially.*

Table 3 shows the relation between the complexity and the severity level of all 213 constraints.

	vocabulary	***simple***	***complex***
info	38.7 %	**76.2 %**	**42.3 %**
warning	6.7 %	7.1 %	13.5 %
error	**54.6 %**	16.7 %	**44.2 %**

Table 3: Complexity vs. Severity of Constraints

Hypothesis 7 *More than the 1/2 of the vocabulary constraints are error constraints, more than 3/4 of the simple constraints are informational constraints, and complex constraints are either informational or error constraints.*

As vocabulary constraints are directly extractable from vocabularies and therefore ensure the syntax and semantics of vocabularies, violations of vocabulary constraints are more severe than violations caused by simple constraints which are easily defined in addition to vocabulary constraints. In many cases, complex constraints are also needed to express severe constraints which cannot be expressed by modeling languages. The relation between complexity and severity shows the importance to develop constraint languages which are enable to express severe complex constraints.

11 Conclusion and Future Work

Thomas: ToDO

[33]

We implemented a validation environment (available at http://purl.org/net/rdfval-demo) to validate RDF data according to constraints expressed my arbitrary constraint languages and to ensure correct syntax and semantics of diverse vocabularies such as *Disco*, *QB*, *SKOS*, and *PHDD* (section 9). We exhaustively evaluated the metadata quality of large real world aggregated (*QB*), person-level (*Disco*), and thesauri (*SKOS*) data sets (more than 4.2 billion triples and 15 thousand data sets) by means of 213 constraints of the majority of the constraint types[34] (section 10).

Thomas: die vokabular-beschreibungen klarer strukturieren. du nutzt plötzlich disco elemente, disco wird überhaupt nicht vernünftig eingeführt, obwohl es später untersucht wird alles, was nicht der aussage des papers (neuer titel, btw) dient, solltest du weglassen klare, einfache struktur: 1. grundidee (constraints identifizieren, validieren auf datensets, zu belastbaren aussagen kommen für die weitere entwicklung von constraint languages) 2. vorstellung der untersuchten vokabulare 3. beschreiben, wo die constraints herkommen, bzw. wie die generiert wurden. dabei auch auf die klassifikation eingehen, aber nur im hinblick auf die spätere auswertung, nicht als main

References

1. Thomas Bosch and Kai Eckert. Requirements on rdf constraint formulation and validation. *Proceedings of the DCMI International Conference on Dublin Core and Metadata Applications (DC 2014)*, 2014.
2. Thomas Bosch and Kai Eckert. Towards description set profiles for rdf using sparql as intermediate language. *Proceedings of the DCMI International Conference on Dublin Core and Metadata Applications (DC 2014)*, 2014.
3. Thomas Bosch, Andreas Nolle, Erman Acar, and Kai Eckert. Rdf validation requirements - evaluation and logical underpinning. 2015.
4. Thomas Bosch, Benjamin Zapilko, Joachim Wackerow, and Kai Eckert. An evaluation of metadata and data quality on person-level, aggregated, thesauri, statistical classifications, and rectangular data sets. 2015.
5. Thomas Bosch, Benjamin Zapilko, Joachim Wackerow, and Kai Eckert. Rdf constraints to validate rectangular data and metadata on person-level data, aggregated data, thesauri, and statistical classifications. 2015.
6. Richard Cyganiak, Simon Field, Arofan Gregory, Wolfgang Halb, and Jeni Tennison. Semantic statistics: Bringing together sdmx and scovo. In Christian Bizer, Tom Heath, Tim Berners-Lee, and Michael Hausenblas, editors, *Proceedings of the*

[33] The first appendix of this paper describing each constraint in detail is available at: http://arxiv.org/abs/1504.04479 [5]

[34] The second appendix of this paper describing the evaluation in detail is available at: http://arxiv.org/abs/1504.04478 [4].

WWW 2010 Workshop on Linked Data on the Web, volume 628 of *CEUR Workshop Proceedings*, 2010.

7. Richard Cyganiak and Dave Reynolds. The rdf data cube vocabulary. W3C recommendation, W3C, January 2014.
8. Carsten Lutz, Carlos Areces, Ian Horrocks, and Ulrike Sattler. Keys, nominals, and concrete domains. *Journal of Artificial Intelligence Research*, 23(1):667–726, June 2005.
9. Christian Mader, Bernhard Haslhofer, and Antoine Isaac. Finding quality issues in skos vocabularies. In *Proceedings of the Second International Conference on Theory and Practice of Digital Libraries*, TPDL'12, pages 222–233, Berlin, Heidelberg, 2012. Springer-Verlag.
10. Michael Schneider. OWL 2 Web Ontology Language RDF-Based Semantics. W3C recommendation, W3C, October 2009.
11. Mary Vardigan, Pascal Heus, and Wendy Thomas. Data documentation initiative: Towards a standard for the social sciences. *International Journal of Digital Curation*, 3(1):107–113, 2008.