Aligning Patterns to the Wikibase Model *

Andrew Eells[1][0000−0001−6357−6646], Cogan Shimizu[1][0000−0003−4283−8701]✉, Lu Zhou[1][0000−0002−0453−9965], Pascal Hitzler[1], Seila Gonzalez[2], and Dean Rehberger[2]

[1] Data Semantics Laboratory, Kansas State University, USA
[2] MATRIX, Michigan State University, USA

Abstract. When developing a knowledge graph, it is important to promote its (re)usability, accessibility, and persistence. One way of accomplishing (re)usability is through the principled use (i.e., using a structured development methodology) of a schema that describes and documents the relations between concepts in the knowledge graph. With respect to accessibility and persistence, one can consider exposing a SPARQL endpoint and allowing interested parties to query against it. While this is a very flexible approach, it makes it difficult to explore the data. On the other hand, one could consider exposing data through a framework such as Wikibase. In this paper, we provide a small library of patterns that link between a traditional ontology design pattern and the underlying Wikibase data model.

1 Introduction

When developing a knowledge graph, there are many aspects to consider during its deployment. These range from usability of its interfaces (both human and programmatic), the (re)usability of the data that it contains, its accessibility (both in terms of uptime and its interfaces), transparency (relating to provenance and trustworthiness), and its persistence. These characteristics are neatly summarized in the FAIR manifesto [10]. One way of accomplishing (re)usability of the data is through the principled use (i.e., using a structured development methodology) of a schema that describes and documents the relations between concepts in the knowledge graph. With respect to accessibility and persistence, one can consider exposing a SPARQL endpoint and allowing interested parties to query against it. While this is a very flexible approach, it makes it difficult to explore the data. On the other hand, one could consider exposing data through a framework such as Wikibase. In this paper, we explore how the Modular Ontology Modeling methodology (MOMo [7]) can be applied in such a way that eventual deployment of the graph data to the Wikibase model is seamless.

Modular Ontology modeling (MOMo) specifies the development of a module for sets of tightly bound key notions that will be included in a given ontology

[7].[3] When developing a module, it is generally suggested to identify applicable *ontology design patterns* (ODPs) [2] and adapt them to the use-case at hand through template-based instantiation [3]. During this process it is good practice to consult existing collections of ODPs, such as the ODP Community Wiki[4] or MODL [8].

As one of the largest publicly editable and accessible knowledge bases, Wikidata is an immense, crowdsourced knowledge base with persistent data that is available for public use and consumption. It would be very difficult to have an ontology of everything, but Wikidata is probably close enough for this purpose. It contains millions of pieces of knowledge from many different domains in the world. In addition, Wikidata serves as the structured data hub for all of Wikimedia's projects (e.g., Wikipedia, Wikivoyage, Wiktionary, and Wikisource). As such, when modeling an ontology, it makes sense to consider ease of integration with resources like Wikidata, among other Linked Data Platforms.[5] *Wikibase* is the underlying technology for the *Wiki-* projects.

The Enslaved Ontology [9] was modeled using the nascent MOMo methodology and is used as the schema for the resulting knowledge base. The Enslaved Hub is an innovative and compelling centralized location for engaging with historical slave trade data from a variety of sources and is supported by an underlying installation of the Wikibase platform hosting the Enslaved Hub.

During development, we had assumed that it would be relatively easy to adapt the modular Enslaved ontology to the Wikibase model. Unfortunately, between different semantics for validating data to be put into the knowledge base and unclear mapping between different notions of provenance, it was not as straightforward as we had expected. In the meantime, we developed some simple and complex mapping patterns between the Enslaved modular ontology and the Wikibase model, established in the Enslaved benchmark [11], which is incorporated into the complex track in Ontology Alignment Evaluation Initiative (OAEI).

This paper takes an alternative approach by starting at the beginning of the modeling process. In particular, by utilizing a library of ontology design patterns that have been specifically engineered to explicitly represent how Wikibase models data "under-the-hood," thus ensuring that the ontology is optimally structured for interoperability with Wikibase. This library can be used by any organization to model their own internal and proprietary knowledge graphs and apply their alignments to Wikibase model as an important tool to augment or induce new information into their own knowledge graph. In particular, this paper provides the first [6] set of such patterns for representing basic modeling problems alongside an intuitive graphical notation of these patterns.

---

[3] We focus on the MOMo paradigm as it is closely aligned with our use case, but any pattern-based methodology, such as eXtreme Design [5] would work similarly.

[4] See `https://ontologydesignpatterns.org/`.

[5] See `https://www.wikidata.org/`.

[6] We are unaware of any existent patterns that address this aspect of modeling with respect to the Wikibase schema.

1.1 The Graphical Syntax

We have developed two layers of notation. The first layer, which we refer to as *abbreviated*, focuses on what an end user would input if they were adding an entry to Wikidata via the web interface. The second layer, referred to as *expanded*, builds on our abbreviated notation by including additional context information that Wikibase adds to statements, depending on their type and content. Our abbreviated notation, seen in figure 1, has the following syntax:

1. Gold boxes with solid borders refer to a Wikibase entity or item.
2. Arrows between nodes denote the type and direction of the relationship. In the diagrams, each relationship will have a prefix, which gives the type of the relationship, followed by a colon, and then the name of the property. The different prefixes are explained in Section 2.
3. Blue ovals denote a specific datatype.
4. Sometimes a property label may emit arrows; these represent context about the relationship. This is explained in detail in Section 3.

Figure 2 shows our *expanded* notation. We follow the syntax from the abbreviated notation, with the following additions:

1. Gold boxes of the form `prefix:<hash>` refer to hash nodes created by Wikibase in order to add context statements to the relationship.
2. Purple boxes are explicit typing information added by Wikibase to give the type of the hash node.
3. Arrows pointing to a purple box are `rdf:type`.

The rest of this paper is organized as follows. Section 2 briefly outlines the standard Wikibase model in order to provide context for the rest of the paper. Section 3 describes our developed patterns with their schema diagrams. Section 4 concludes with future work.

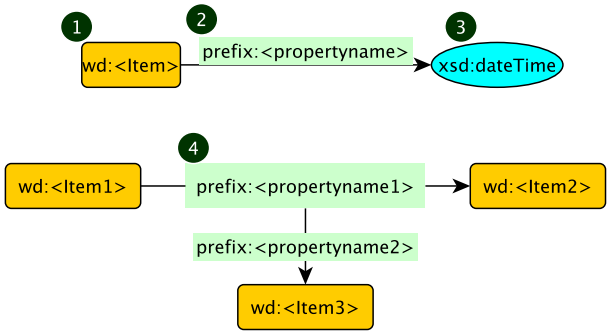


Fig. 1: Abbreviated Notation

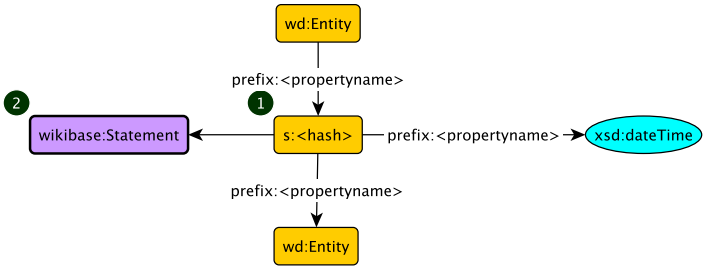


Fig. 2: Expanded Notation

2 The Wikibase Model

A Wikibase knowledge base is a collection of entities. Entities are the basic elements of the knowledge base, which can be described and referenced using the Wikibase data model. The Wikibase data model describes the structure of the data that is handled in Wikibase. In particular, it specifies which kind of information users can contribute to the system. In this section, we introduce how Wikibase models data in RDF format and some basic concepts that will be used in the rest of the paper.

There are several predefined concepts in Wikibase data model introduced as follows:

- **Items:** Items are the way Wikibase refers to anything of interest. For every Item, there is some basic information that clarifies what the Item is about, such as the link to a Wikipedia page in some language. Each Item has a unique identifier of the format `Q<number>`, with a prefix of `wd:`, which is abbreviation of `http://www.wikidata.org/entity/`, such as `wd:Q1234` in figure 3. There are also human readable labels and short descriptions that are used to help Wikidata users find the right Item.
- **Statements:** Statements are descriptions that users have entered about the Item. The RDF format represents statements in two forms. The first type is named as a **Truthy statement**. These are simple triples that assert facts. The other statement type is the **Full statement**, which is used to represent all data about the statement in the system. To differentiate them, Wikidata usually uses specific namespaces in specific places in both truthy and full statements. For example, in figure 3, the namespace `http://www.wikidata.org/prop/direct/`, which is abbreviated using the prefix `wdt:`, is usually used for the predicate of a truthy statement, and the range of the predicate is the simple value for the statement. The full statement is represented as separate node using property reification, with prefix `wds:` (`http://www.wikidata.org/entity/statement/`) with the ID of the statement (e.g., `wds:12345678` in figure 3). There is no guaranteed format or meaning to the statement id. The statements are linked to the entity with

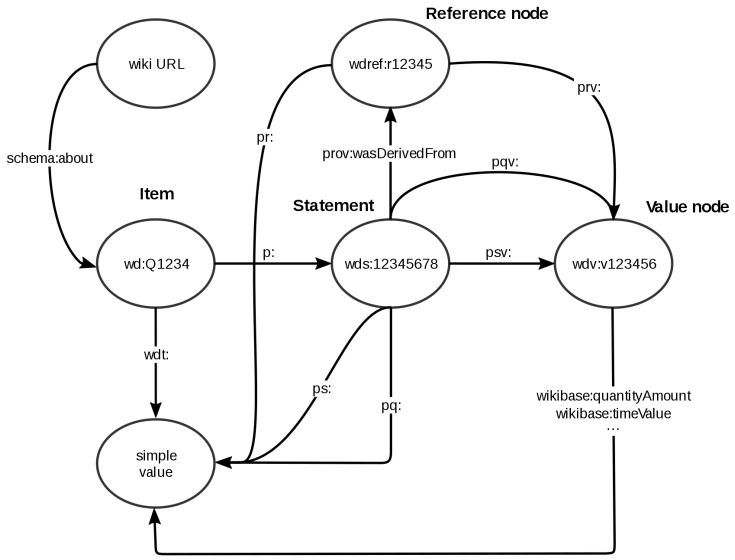


Fig. 3: Wikidata Model [1]

the predicate with prefix `p:` (`http://www.wikidata.org/prop/`) and the name of the property. The simple value is represented by the predicate with prefix `ps:` (`http://www.wikidata.org/prop/statement/`) and the name of the property while the full value is represented by the predicate with prefix `psv:` (`http://www.wikidata.org/prop/statement/value/`).

– **Qualifiers:** The statement always has no more than one value, but can have multiple qualifiers and references. The qualifiers are used to further describe or refine the value of a property given in a statement. They are represented by predicates with prefix `pq:` (`http://www.wikidata.org/prop/qualifier/`), and the object is the simple value of the qualifier. The full value is represented by the predicate with prefix `pqv:` (`http://www.wikidata.org/prop/qualifier/value/`), and the range of the predicate is the full value node.

– **References:** References offer a source that supports the given claim. They are represented by the predicate `prov:wasDerivedFrom` with the object being the reference node. Similar to qualifiers, the simple value of the reference is represented by predicates with prefix `pr:` (`http://www.wikidata.org/prop/reference/`), while the full value is represented by the predicate with prefix `prv:` (`http://www.wikidata.org/prop/reference/value/`).

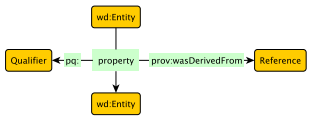
3 The Patterns

In what follows we outline the patterns we have designed. In each pattern diagram, we follow the notation syntax outlined in Section 1.1, and shown in figure 1 and figure 2. For each pattern, we show our two notations:

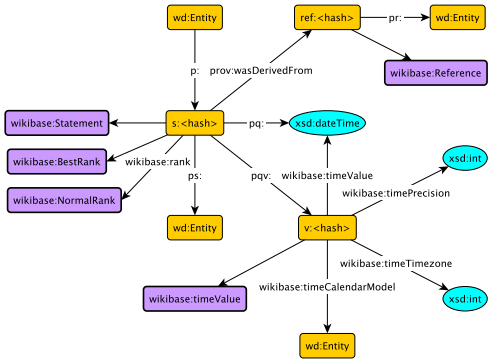
- Our abbreviated notation shows what an end user might provide in the front-end interface of Wikibase if they were to add an item of that datatype (string, dateTime, quantity, etc.) and usage (statement value vs qualifier).
- Our expanded notation shows the additional context information that Wikibase adds to the given datatype/usage combination.

The patterns we have provided here are only a starting point. Wikidata and Wikibase offer native support for an ever-growing list of datatypes, and even more can be defined with various extensions. We have chosen to focus initially on three core datatypes that are defined in OWL [4], and offer the most general usability and greatest impact.

3.1 Statement with Qualifier and Reference



(a) Abbreviated



(b) Expanded

Fig. 4: Statement with Qualifier and Reference

The essential components of Wikibase, and what makes it an intensely powerful platform for knowledge management, are the statements that describe each entity. Each statement makes a claim about the entity, and may or may not have a qualifier - which provides additional information about the claim (such as when the claim was valid) - or a reference (where the claim was sourced from). These statements link and describe entities in a structured way that we have modeled in figure 4.

The abbreviated model for this (figure 4a) is fairly straight forward. We have an entity, seen here in the gold box containing `wd:Entity`, linked by a property shown in the green box to another entity. Context information is then provided for that specific Entity-Property-Entity relationship, which may include one or more qualifiers or references. Each qualifier is linked to the statement by a `pq:PXXX` identifier, with the prefix `pq:` here denoting that the relationship is that of a property qualifier. References are linked via `prov:wasDerivedFrom` which will be expanded upon below.

The expanded model, seen in figure 4b, shows not only the information described above, but also yet more context information provided by Wikibase to each statement. Starting again from our main `wd:Entity` node, we see it liked by a property identified by a property ID with the format `p:PXXX` to a hashed node, identified by `s:<hash>`. For the property, the prefix `p:` denotes that the type of the relationship is that of a property, while the `PXXX` provides the property ID. On the hashed statement node, `s:` here denotes that the node is a statement. Coming off of the `s:<hash>` node, we see the link to another `wd:Entity` via a property statement with the format `ps:PXXX`. As with the abbreviated diagram, qualifiers are linked via `pq:PXXXX`.

The three nodes on the left, prefixed with `wikibase:` are information added by Wikibase to describe the statement. The first, `wikibase:Statement` types the hashed node as a statement, rather than a qualifier or a reference. The other two, `wikibase:BestRank` and `wikibase:NormalRank` provide the style of ranking for the statement, and denote whether or not this statement is the "best" statement for this Entity-Property-Entity relationship. For example, if an entity is linked to two other entities, both by the same property, one of them can be ranked higher than the other, denoting it as the "best" or preferred value for the statement.

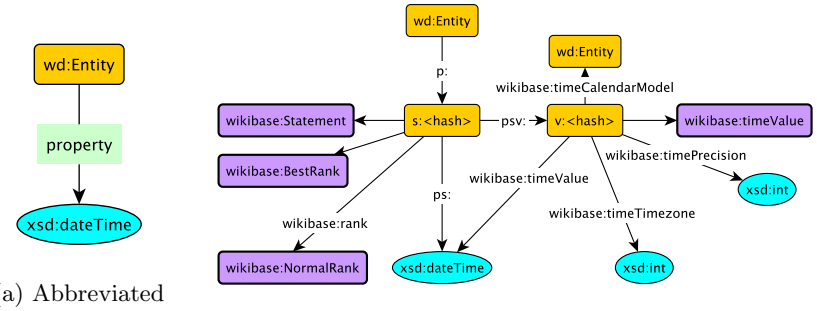
If a reference is provided for a given statement, Wikibase creates another hashed node linked via `prov:wasDerivedFrom` [6], shown with the format `ref:<hash>`. The prefix `ref:` denotes the node as a reference node, which is also denoted by the `wikibase:Reference` node coming off of the `ref:<hash>`. Lastly, the `ref:<hash>` node is linked to the corresponding entity for the reference via `pr:PXXX`.

The qualifier example in figure 4b, as with many datatype values, has a complex structure that we will elaborate on in the next section.

3.2 The DateTime Pattern

In the patterns that follow, we will see that there is some significance to whether a given datatype is used as the value of a statement or a qualifier. Let's first look at the use of a `dateTime`, shown in figures 5 and 6.

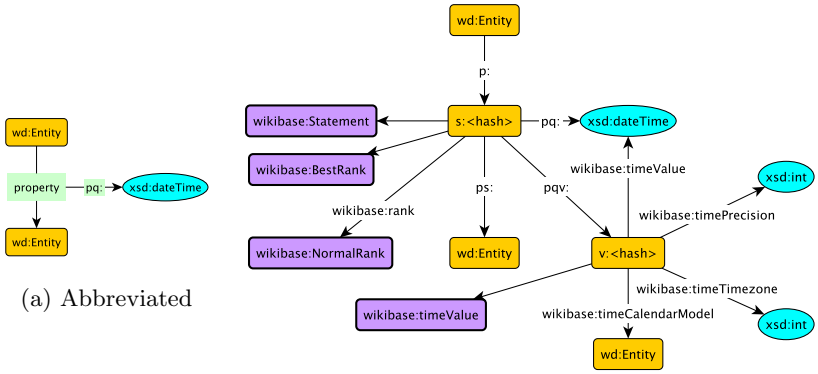
The abbreviated notation is quite straightforward indeed. As a statement value, an entity is linked to a dateTime value via a property. But, as shown in figure 5b and as mentioned in section 3.1 above, Wikibase adds several data points to this that make each statement much more useful. In addition to the information added to the `s:<hash>` node, including a simple representation of the value via `ps:PXXX`, Wikibase adds another hashed node. This time the added



(a) Abbreviated

(b) Expanded

Fig. 5: dateTime as Statement Value



(a) Abbreviated

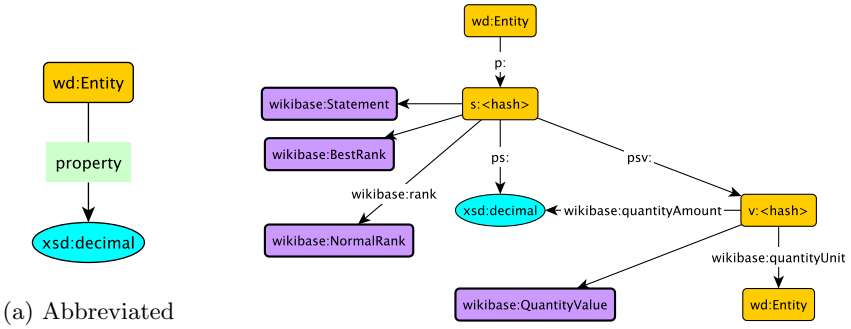
(b) Expanded

Fig. 6: dateTime as Qualifier

node will house the additional context information about the value. The statement node is linked to the `v:<hash>` value node via `psv:PXXX`, with the prefix `psv:` denoting the relationship as that of a property statement value.

Wikibase types the value node explicitly via `wikibase:timeValue`. In this instance, given that the datatype is a dateTime value, Wikibase adds a link to the Wikidata entity for the calendar model used (i.e. Gregorian, etc.) via `wikibase:timeCalendarModel`. Also provided is a number representing the timezone in offset from UTC, via `wikibase:timeTimezone`, and a number representing the time precision via `wikibase:timePrecision`.[7] Lastly the value node links back to the simple value for the dateTime via `wikibase:timeValue`.

[7] There are 15 levels of time precision on Wikibase, with 0 corresponding to "billions of years", and 14 corresponding to seconds.



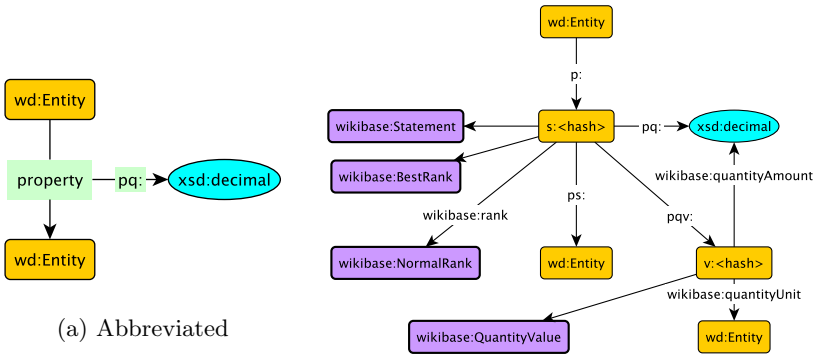
(a) Abbreviated

(b) Expanded

Fig. 7: Quantity as Statement Value

When a `dateTime` is used in a qualifier, the internal structure for the `v:<hash>` node remains unchanged. However, rather than linking to the statement hash node via `ps:PXXX` and `psv:PXXX`, it is via `pq:PXXX` and `pqv:PXXX` respectively.

3.3 Quantity



(a) Abbreviated

(b) Expanded

Fig. 8: Quantity as Qualifier

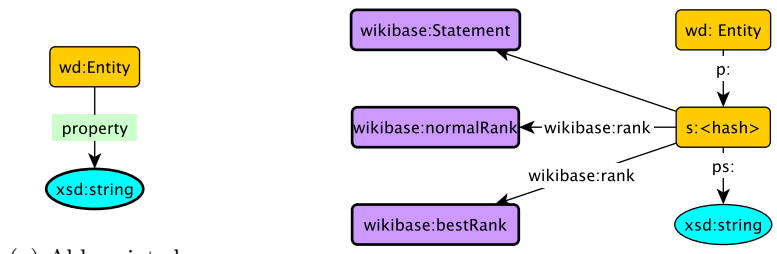
The quantity pattern (figures 7 and 8) mimics some of the features of dateTime, but Wikibase then adds some specific data about the quantity mentioned. When the quantity is being used as the value of a statement, the abbreviated notation continues to be quite simple. A `wd:Entity` is linked to a `xsd:decimal` via a property.

Expanding that model out, we see that Wikibase again adds a hashed node (`v:<hash>`). The statement hash links to the simple value for the quantity via `ps:PXXX` and to the value node via `psv:PXXX`.

When the quantity is a qualifier, our abbreviated notation (figure 6a) shows that the quantity is linked to from the property via `pq:PXXX`. In its expanded form (figure 6b), Wikibase creates the same `v:<hash>` node mentioned previously however it links to the corresponding nodes via `pq:PXXX` and `pqv:PXXX`.

Wikibase adds two datapoints for a quantity, whether it is a statement value or a qualifier. Firstly, it types the hash node via `wikibase:QuantityValue`. It also links to the Wikidata entity for the unit of measurement for that quantity via `wikibase:quantityUnit`. Finally, the value node links back to the simple value of the quantity via `wikibase:quantityAmount`.

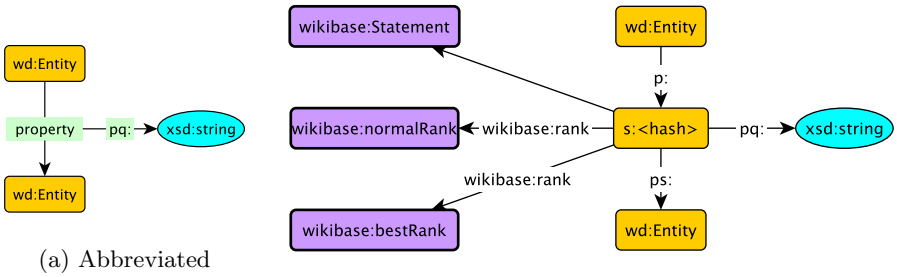
3.4 String



(a) Abbreviated

(b) Expanded

Fig. 9: String as Statement Value



(a) Abbreviated

(b) Expanded

Fig. 10: String as Qualifier

In a slight deviation from the previous datatypes, Wikibase only adds ranking (`wikibase:rank`) and typing (`wikibase:Statement`) information for a statement with a string (figures 9 and 10).

As the abbreviated notation shows, when used as the value of a statement, an entity is linked to a string via a property. In the case of a string being used as a qualifier, a property is linked to a string via `pq:PXXX`.

Notably, no value node is created for a string.

4 Conclusion

When developing and deploying a knowledge graph, there are many obstacles to a persistent, transparent, and usable resource. One way to overcome these obstacles is to use the Wikibase framework. In this paper, we have represented several common modeling constructions in a graphical syntax that makes it clear how they map into the Wikibase context. This should allow ontology developers to more quickly, accurately, and with reduced effort create ontologies (or knowledge graph schema) that are "Wikibase ready," thus improving persistence and accessibility of the deployed knowledge graph. With a base understanding of how these frequently occurring design patterns are represented in the Wikibase model, there are many potential next steps. Of particular importance and urgency would be

1. the complete translation of MODL [8] patterns;
2. the complete translation of Enslaved Ontology patterns.

Regarding the former, this would facilitate the creation of arbitrary, modular ontologies that are intended to be deployed to the Wikibase framework, thus encouraging a persistent (if Wikidata is utilized) and usable deployment of the knowledge graph in question. For the latter, we will be able to examine the efficacy of our approach by comparing the re-implementation of the Enslaved Ontology patterns to the manually altered modules utilized in the Enslaved Portal (see `https://enslaved.org/`).

Acknowledgements. The authors acknowledge support by the National Science Foundation under Grant 2032628 *EAGER: Open Science in Semantic Web Research* and Grant 2033521 A1: KnowWhereGraph: Enriching and Linking Cross-Domain Knowledge Graphs using Spatially-Explicit AI Technologies, as well as the Mellon Foundation through the *Enslaved: Peoples of the Historic Slave Trade*.

References

1. Wikibase/indexing/rdf dump format - mediawiki (Mar 2021), `https://www.mediawiki.org/wiki/Wikibase/Indexing/RDF_Dump_Format`
2. Gangemi, A., Presutti, V.: Ontology design patterns. In: Staab, S., Studer, R. (eds.) Handbook on Ontologies, pp. 221–243. International Handbooks on Information Systems, Springer (2009). https://doi.org/10.1007/978-3-540-92673-3_10, `https://doi.org/10.1007/978-3-540-92673-3_10`

3. Hammar, K., Presutti, V.: Template-based content ODP instantiation. In: Hammar, K., Hitzler, P., Krisnadhi, A., Lawrynowicz, A., Nuzzolese, A.G., Solanki, M. (eds.) Advances in Ontology Design and Patterns [revised and extended versions of the papers presented at the 7th edition of the Workshop on Ontology and Semantic Web Patterns, WOP@ISWC 2016, Kobe, Japan, 18th October 2016]. Studies on the Semantic Web, vol. 32, pp. 1–13. IOS Press (2016). https://doi.org/10.3233/978-1-61499-826-6-1, https://doi.org/10.3233/978-1-61499-826-6-1
4. Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P.F., Rudolph, S. (eds.): OWL 2 Web Ontology Language Primer (Second Edition. W3C Recommendation 11 December 2012 (2012), available from http://ww.w3.org/TR/owl2-primer/
5. Presutti, V., Daga, E., Gangemi, A., Blomqvist, E.: eXtreme Design with content ontology design patterns. In: Blomqvist, E., Sandkuhl, K., Scharffe, F., Svátek, V. (eds.) Proceedings of the Workshop on Ontology Patterns (WOP 2009) , collocated with the 8th International Semantic Web Conference (ISWC-2009), Washington D.C., USA, 25 October, 2009. CEUR Workshop Proceedings, vol. 516. CEUR-WS.org (2009), http://ceur-ws.org/Vol-516/pap21.pdf
6. Sahoo, S., McGuinness, D., Lebo, T.: PROV-o: The PROV ontology. W3C recommendation, W3C (Apr 2013), http://www.w3.org/TR/2013/REC-prov-o-20130430/
7. Shimizu, C., Hammar, K., Hitzler, P.: Modular ontology modeling. Tech. rep. (2021), https://daselab.cs.ksu.edu/publications/modular-ontology-modeling, under Review.
8. Shimizu, C., Hirt, Q., Hitzler, P.: MODL: A modular ontology design library. In: Janowicz, K., Krisnadhi, A.A., Poveda-Villalón, M., Hammar, K., Shimizu, C. (eds.) Proceedings of the 10th Workshop on Ontology Design and Patterns (WOP 2019) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 27, 2019. CEUR Workshop Proceedings, vol. 2459, pp. 47–58. CEUR-WS.org (2019), http://ceur-ws.org/Vol-2459/paper4.pdf
9. Shimizu, C., Hitzler, P., Hirt, Q., Shiell, A., Gonzalez, S., Foley, C., Rehberger, D., Watrall, E., Hawthorne, W., Tarr, D., Carty, R., Mixter, J.: The enslaved ontology 1.0: People of the historic slave trade. Tech. rep., Michigan State University, East Lansing, Michigan (April 2019)
10. Wilkinson, M.D., Dumontier, M., et al.: The fair guiding principles for scientific data management and stewardship. Scientific Data **3**, 160018 EP – (Mar 2016), https://doi.org/10.1038/sdata.2016.18, comment
11. Zhou, L., Shimizu, C., Hitzler, P., Sheill, A.M., Estrecha, S.G., Foley, C., Tarr, D., Rehberger, D.: The enslaved dataset: A real-world complex ontology alignment benchmark using wikibase. In: d'Aquin, M., Dietze, S., Hauff, C., Curry, E., Cudré-Mauroux, P. (eds.) CIKM '20: The 29th ACM International Conference on Information and Knowledge Management, Virtual Event, Ireland, October 19-23, 2020. pp. 3197–3204. ACM (2020). https://doi.org/10.1145/3340531.3412768, https://doi.org/10.1145/3340531.3412768