

Automating the use of Shape Expressions for the validation of semantic knowledge in Wikidata

Houcemeddine Turki^{1,*}, Mohamed Ali Hadj Taieb¹, Khalil Chebil¹, Mohamed Ben Aouicha¹, Lane Raspberry² and Daniel Mietchen^{3,4}

¹Data Engineering and Semantics Research Unit, Faculty of Sciences of Sfax, University of Sfax, Sfax, Tunisia

²School of Data Science, University of Virginia, Charlottesville, VA, United States of America

³Ronin Institute for Independent Scholarship, Montclair, New Jersey, United States of America

⁴FIZ Karlsruhe – Leibniz Institute for Information Infrastructure, Berlin, Germany

Abstract

In this position paper, we discuss the semantic alignment-based approach for automating the ShEx-based validation of Wikidata items as proposed by Wikimedia Deutschland in July 2023, and we propose an alternative method that automates the shape-based validation of Wikidata entities and statements based on the conversion of ShEx EntitySchemas into SPARQL queries that identify relevant entities. We explain the advantages and drawbacks of both methods to provide the community with a useful overview of the matter.

Keywords

Wikidata, Knowledge Graph Validation, Shape Expressions, SPARQL, Semantic Alignment, Data Quality

1. Introduction


With the rise and growth of open knowledge graphs, ensuring their quality becomes increasingly challenging [1]. Particularly, quality assessment has been a vital component of the development of Wikidata as an open and collaborative knowledge graph, as it follows the same principles as Wikipedia, including its quest for consistency and completeness [2]. Subsequently, Wikidata has supported the creation of EntitySchemas implemented in Shape Expressions (ShEx) to ensure the shape-based validation of the Wikidata items [3], following preliminary experiments conducted in 2019 [4]. Currently, there are multiple EntitySchemas to validate a number of Wikidata classes¹, particularly the ones related to the molecular biology field [3]. Several efforts have also been directed toward creating tools for the automatic generation of EntitySchemas, like *sheXer* [3]. There are even several adaptations of the Shape Expressions (ShEx) language to

Wikidata'23: Wikidata Workshop at ISWC 2023

*Corresponding author.

✉ turkiabdelwaheb@hotmail.fr (H. Turki); mohamedali.hajtaieb@fss.usf.tn (M. A. Hadj Taieb); khalil.chebil@insat.ucar.tn (K. Chebil); mohamed.benaouicha@fss.usf.tn (M. Ben Aouicha); lr2ua@virginia.edu (L. Raspberry); daniel.mietchen@ronininstitute.org (D. Mietchen)

ORCID 0000-0003-3492-2014 (H. Turki); 0000-0002-2786-8913 (M. A. Hadj Taieb); 0000-0002-7571-4150 (K. Chebil); 0000-0002-2277-5814 (M. Ben Aouicha); 0000-0002-9485-6146 (L. Raspberry); 0000-0001-9488-1870 (D. Mietchen)

 © 2023 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

 CEUR Workshop Proceedings (CEUR-WS.org)

¹A list of the EntitySchemas available in Wikidata can be found at https://www.wikidata.org/wiki/Wikidata:Database_reports/EntitySchema_directory.

make it more intuitive for Wikidata users to write EntitySchemas, like *ShExStatements* [5] and *WShEx* [6]. Furthermore, there are several initiatives to combine ShEx with SPARQL to enhance its scope beyond shape-based validation [7]. Yet, there is limited progress toward automating the use of ShEx EntitySchemas for the validation of Wikidata entities. A new initiative of Wikimedia Deutschland aims to solve this problem based on creating semantic alignments between ShEx EntitySchemas and Wikidata Classes.

In this position paper, we describe the current situation of the ShEx-based validation of semantic knowledge in Wikidata (Section 2). Then, we outline the approach of Wikimedia Deutschland for automating the ShEx-based validation of Wikidata (Section 3). After that, we propose an alternative approach for the ShEx-based validation of Wikidata driven by the conversion of ShEx EntitySchemas into corresponding SPARQL queries (Section 4). Later, we discuss the strengths and limitations of both approaches to give the Wikidata community different perspectives on how to handle this issue (Section 5). Finally, we draw conclusions and propose future directions for this research work (Section 6).

2. ShEx-based validation of semantic knowledge in Wikidata

In this section, we delve into the utilization of Shape Expressions (ShEx) for the validation of semantic knowledge in Wikidata. ShEx provides a structured approach to assess the conformity of Wikidata entities to predefined schemas. These schemas, known as EntitySchemas, define the expected structure and constraints for different classes of entities. The integration of ShEx into Wikidata's data quality control framework introduces a standardized method for ensuring data accuracy and consistency. The main preconditions for using ShEx to validate Wikidata entries are as follows:

- **EntitySchemas:** EntitySchemas are at the core of ShEx-based validation. They define the structural expectations and constraints for specific classes of entities within Wikidata. Currently, Wikidata has approximately 100 million items spanning several million classes. However, there are only around 400 EntitySchemas available. These schemas cover a wide spectrum of classes, from highly populated ones like "human" (E10) and "city" (E100) to less-represented ones like "Nazca lines" (E148) and "ethics committees" (E396). The level of detail and complexity of these schemas varies significantly. Some employ direct constraints, which are single straightforward one-line constraints, while others use open constraints, where the object is a variable, and indirect constraints, which are sets of chained constraints where the object of one constraint is the subject of another constraint. There are also closed constraints, which define constraints where the object is a predefined set of items (See the EntitySchema for a deceased person [E105] for examples of every type of condition, as per Figure 1). Some EntitySchemas even rely on other schemas through the use of IMPORT clauses to define the entities corresponding to them and to provide further constraints on how entities should be defined (e.g., E192 for a virus taxon refers to E69 when describing diseases that the virus might be involved in). This diversity reflects the heterogeneous nature of Wikidata's data structure.
- **Identifying Entities:** To apply a specific schema to a group of entities, Wikidata relies on SPARQL queries. Schema documentation often includes sample SPARQL queries that

can be executed using tools like the Wikidata Query Service or RDF dump-based external SPARQL endpoints [8]. These queries identify the entities subject to validation based on specified criteria, allowing for a focused validation process.

- **Validation Tools:** The validation process itself is facilitated by dedicated tools. On each EntitySchema’s Wikidata page, there is a “check entities against this Schema” hyperlink [4]. This link directs users to the Simple Online Validator², a tool hosted on the Wikimedia Toolforge infrastructure. The validator consumes the EntitySchema, executes SPARQL queries to retrieve relevant entities, applies the schema’s constraints, and reports compliance or non-compliance, providing detailed feedback when necessary. Additionally, other validation tools and methods, such as those for identifying subsets of Wikidata [9], are available to users.
- **Signaling Compliance:** Presently, the signaling of ShEx schema compliance to Wikidata users is somewhat limited. While dedicated validation tools report compliance or non-compliance, this information is not prominently integrated into the SPARQL endpoint or the graphical user interface (GUI). Nonetheless, Wikidata employs various non-ShEx quality control mechanisms [4], such as constraint statements and bot-curated pages, to identify and report data quality issues [10]. Adapting these workflows to incorporate ShEx-based compliance reporting remains a possibility.

ShEx schemas provide a structured framework for assessing data quality in Wikidata, yet several challenges and considerations persist. One fundamental concern revolves around the coverage of EntitySchemas in relation to the vast number of entities within Wikidata. It is crucial to ascertain the extent to which the 400 schemas encompass Wikidata’s diverse entity classes. It is conceivable that these schemas predominantly address well-represented classes, potentially leaving a long tail of classes with limited coverage. Additionally, the subjective nature of schema definitions, particularly for highly specific or niche classes, may have contributed to the slow adoption of ShEx in Wikidata.

Another impediment to the widespread adoption of ShEx in Wikidata has been the absence of automated tools for ShEx-based validation. In the subsequent sections, we will introduce two solutions to address this issue. The first solution, proposed by Wikimedia Deutschland, leverages semantic alignments between Wikidata classes and EntitySchemas. The second solution, presented in this position paper, revolves around the transformation of ShEx EntitySchemas into corresponding SPARQL queries through a rule-based approach.

3. The Wikimedia Deutschland solution: Semantic alignment between Wikidata classes and EntitySchemas

In July 2023, the development team of Wikimedia Deutschland deployed a new property in test Wikidata³, the sandbox for Wikidata-related experiments, to assign EntitySchemas to their corresponding Wikidata classes. This property is called *EntitySchema for this class*⁴ and has

²<https://shex-simple.toolforge.org/wikidata/>

³<https://test.wikidata.org>.

⁴<https://test.wikidata.org/wiki/Property:P97725>.

already been discussed by the community since 28 May 2019⁵. The Proposal implies the creation of a new datatype for EntitySchemas and proposes to align the Wikidata classes as subjects to EntitySchemas as objects of the *EntitySchema for this class* relations, as shown in Figure 2. This will allow the development of tools that process the taxonomic relations of a given item in Wikidata (i.e., *instance of* [P31], *subclass of* [P279], and *part of* [P361]) to identify the EntitySchemas that are relevant to use for checking the consistency of the considered entity.

4. Our solution: SPARQL-based identification of relevant items

What we propose is to analyze the EntitySchema itself to identify the Wikidata items that are relevant to it. This should be enabled by converting the ShEx statements into a SPARQL query that can be used to retrieve the Wikidata items that should be considered. This solution has been proposed since the early days of ShEx in 2017⁶. The principle is based on using closed constraints to create a SPARQL query to find the Wikidata items corresponding to the considered EntitySchema. By a closed constraint, we mean the chain of ShEx statements having a definite set of Wikidata items (beginning with *wd:Q*) as a final object, as shown in Figure 1.

As SPARQL serves as the query language for RDF knowledge graphs, and ShEx functions as the semantic web language for shape-based validation of knowledge graphs, it's important to note that these two languages do not share identical syntax or structure [10]. Therefore, when dealing with closed constraints, it becomes necessary to undergo a conversion process, transforming them into SPARQL statements prior to their utilization in retrieving specific items from Wikidata. This conversion process is illustrated in Figure 3. In Figure 3, we can observe this conversion process in action, employing the example of E50, which represents the Wikidata EntitySchema for national flags. Additionally, the figure demonstrates the effectiveness of this approach using the schema for genes or variants with references (E390), thereby proving the versatility of this method across EntitySchemas, regardless of their complexity or scope.

A preliminary edition of the source code that can convert EntitySchemas into SPARQL queries is currently available at <https://github.com/csisc/WikidataShExSPARQL>. The processing of closed constraints involves the parsing of curly brackets using the *parentheses level count algorithm* to assign opening curly brackets to their corresponding closing curly brackets [11], the elimination of several non-alphanumeric characters (e.g., *, #, and +), and the removal of EXTRA properties if directly following a variable name. Later, a set of rules will be used to transform the remaining skeleton of the ShEx EntitySchema into a SPARQL query, as shown in Table 1. The obtained SPARQL query will be run through the SPARQL endpoint of Wikidata (<https://query.wikidata.org>) to identify all the Wikidata items that can be validated using the considered EntitySchema.

5. Discussion

The method proposed by Wikimedia Deutschland introduces a valuable approach to aligning Wikidata classes with EntitySchemas. However, it is essential to acknowledge its limitations,

⁵https://www.wikidata.org/wiki/Wikidata:Property_proposal/Shape_Expression_for_class.

⁶<https://github.com/shexSpec/shex/issues/75>.

Table 1
Rules for ShEx-SPARQL conversion

Rule	ShEx	SPARQL
1	$\langle v_1 \rangle \{ p_1 o_1; p_2 o_2; \}$	$?v_1 p_1 o_1; p_2 o_2.$
2	$\langle v_1 \rangle \{ p1_1 \{ p1_2 o_1; \} \}$	$?v_1 p1_1 [p1_2 o_1].$
3	$\langle v_1 \rangle \{ p_1 [o_1 o_2]; \}$	VALUES ?o {o ₁ o ₂ } $?v_1 p_1 ?o.$
4	@ $\langle v_1 \rangle$	$?v_1$
5	start = @ $\langle v_1 \rangle$ @ $\langle v_2 \rangle$ $\langle v_1 \rangle \{ \text{Statements for } v_1 \}$ $\langle v_2 \rangle \{ \text{Statements for } v_2 \}$	SELECT ?id WHERE { {?id Statements for v ₁ } UNION {?id Statements for v ₂ } }
6	$\langle v_1 \rangle \{ p_1 \text{ EXTRA } p_2 o_1 \}$	$?v_1 p_1 [p_2 o_1].$
7	PREFIX <i>pre</i> : $\langle uri \rangle$	PREFIX <i>pre</i> : $\langle uri \rangle$

particularly when EntitySchemas are not directly related to Wikidata classes. For instance, consider an EntitySchema designed for *Tunisian scientists*. In many cases, Wikidata items within this category may not explicitly assign *Tunisian scientist* as the object of an *instance of* [P31] or *facet of* [P1269] statement. Instead, such items are often represented as a combination of properties like {"occupation", "scientist"} (P106, Q901) and {"country of citizenship", "Tunisia"} (P27, Q948). Adding items like "Tunisian scientist" as objects of "facet of" [P1269] relations could result in an influx of redundant data without enriching the underlying semantic knowledge, which is inadvisable given the growing size of Wikidata and its data storage challenges [12]. Another potential solution is the use of reification to specify requirements for an item to be considered beyond class membership [7], as illustrated in Figure 4. The same solution can be applied by using the "category contains" [P4224] property to specify how class members should be defined in Wikidata, as in <https://www.wikidata.org/wiki/Q6471216#P4224> [13]. However, this approach may introduce additional complexity when developing tools based on "EntitySchema for this class" statements to identify corresponding EntitySchemas for a Wikidata item [7].

Regarding our proposed solution, which aims to generate SPARQL queries to identify Wikidata items corresponding to ShEx EntitySchemas, it offers flexibility by implicitly inferring the conditions for item inclusion without the need to store them as RDF triples [14]. The formulation of constraints as SPARQL queries provides adaptability, making it possible to identify corresponding items regardless of the complexity of the constraints [15]. While the generated SPARQL queries are primarily intended for retrieving the set of Wikidata items corresponding to an EntitySchema (i.e., *subsetting*) [14], a slight adaptation of the query can efficiently verify whether an item meets the specified criteria (as depicted in the red query in Figure 5). If an item does not conform to the query, it will return an empty result; if it does, it will return the Wikidata ID of the item as a result.

Our method may have the drawback of increasing the computational stress on the Wikidata Query Service (<https://query.wikidata.org>). The SPARQL endpoint of Wikidata has faced scalability challenges, leading to several outages since 2020⁷. Any additional workload imposed on the Wikidata Query Service may exacerbate its current situation. In contrast, the approach proposed by Wikimedia Deutschland does not rely on advanced computations, making it an attractive option.

In terms of recall, both approaches have inherent limitations. The proposed methods rely on the availability and accuracy of data within Wikidata. If a schema imposes highly specific constraints that are not fully represented in Wikidata entries, there may be entities that go unidentified. For instance, identifying a "Tunisian scientist" would be challenging if the occupation and nationality are not explicitly documented in Wikidata entries. While these limitations are challenging to completely mitigate, they should be considered when implementing ShEx-based validation methods in Wikidata and when interpreting the results. Further research and refinement of these methods may help enhance their recall capabilities in the future.

6. Conclusion

In this position paper, we explain the current efforts for automating the shape-based validation of Wikidata items based on ShEx EntitySchemas and we propose our preliminary solution for this matter that is driven by converting ShEx EntitySchemas into SPARQL queries to identify relevant items. We have shown that our solution is more efficient than the one proposed by *Wikimedia Deutschland*, as it requires less data storage and supports complex constraints that cannot be dealt with using semantic alignments between Wikidata items and ShEx EntitySchemas. Although our solution is more practical, it requires an upgrade to overcome its limitations caused by the performance limits of the Wikidata Query Service. As a future direction of this work, we propose to optimize our source code for efficiency by adjusting its layout and substituting the use of the Wikidata Query Service with other methods for mining Wikidata, like *CirrusSearch*⁸.

Acknowledgments

This research is funded by the Wikimedia Research Fund of Wikimedia Foundation (San Francisco, California, United States of America) through the *Adapting Wikidata to support clinical practice using Data Science, Semantic Web and Machine Learning* Project.⁹ Source code is made available under the MIT License at <https://github.com/csisc/WikidataShExSPARQL>.

⁷See https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service/WDQS_backend_update for further details.

⁸<https://www.mediawiki.org/wiki/Extension:CirrusSearch>

⁹https://meta.wikimedia.org/wiki/Research:Adapting_Wikidata_to_support_clinical_practice_using_Data_Science,_Semantic_Web_and_Machine_Learning

References

- [1] K. Shenoy, F. Ilievski, D. Garijo, D. Schwabe, P. Szekely, A study of the quality of Wikidata, *Journal of Web Semantics* 72 (2022) 100679. doi:10.1016/j.websem.2021.100679.
- [2] A. Piscopo, E. Simperl, What we talk about when we talk about Wikidata quality: a literature survey, in: *Proceedings of the 15th International Symposium on Open Collaboration*, ACM, 2019. doi:10.1145/3306446.3340822.
- [3] A. Waagmeester, E. L. Willighagen, A. I. Su, M. Kutmon, J. E. L. Gayo, D. Fernández-Álvarez, Q. Groom, P. J. Schaap, L. M. Verhagen, J. J. Koehorst, A protocol for adding knowledge to Wikidata: aligning resources on human coronaviruses, *BMC Biology* 19 (2021). doi:10.1186/s12915-020-00940-y.
- [4] K. Thornton, H. Solbrig, G. S. Stupp, J. E. L. Gayo, D. Mietchen, E. Prud'hommeaux, A. Waagmeester, Using Shape Expressions (ShEx) to Share RDF Data Models and to Guide Curation with Rigorous Validation, in: *The Semantic Web*, Springer International Publishing, 2019, pp. 606–620. doi:10.1007/978-3-030-21348-0_39.
- [5] J. Samuel, ShExStatements: Simplifying Shape Expressions for Wikidata, in: *Companion Proceedings of the Web Conference 2021*, ACM, 2021. doi:10.1145/3442442.3452349.
- [6] J. E. Labra-Gayo, Wshex: A language to describe and validate wikibase entities, in: *Proceedings of the 3rd Wikidata Workshop*, 2022, p. 3. URL: <https://ceur-ws.org/Vol-3262/paper3.pdf>.
- [7] H. Turki, M. A. H. Taieb, T. Shafee, T. Lubiana, D. Jemielniak, M. B. Aouicha, J. E. L. Gayo, E. A. Youngstrom, M. Banat, D. Das, D. Mietchen, on behalf of WikiProject COVID, Representing COVID-19 information in collaborative knowledge graphs: The case of Wikidata, *Semantic Web* 13 (2022) 233–264. doi:10.3233/sw-210444.
- [8] D. Hernández, A. Hogan, C. Riveros, C. Rojas, E. Zerega, Querying wikidata: Comparing SPARQL, relational and graph databases, in: *Lecture Notes in Computer Science*, Springer International Publishing, 2016, pp. 88–103. URL: https://doi.org/10.1007/978-3-319-46547-0_10. doi:10.1007/978-3-319-46547-0_10.
- [9] J. E. Labra-Gayo, A. C. González Cavazos, A. Waagmeester, N. Hofmann, S. A. Hosseini Beghaeiraveri, E. Prud'hommeaux, S. Ul-Hasan, E. Willighagen, A. Ammar, Enhancement and reuse of biomedical knowledge graph subsets, *BioHackrXiv* (2022). URL: <https://doi.org/10.37044/osf.io/n7qku>. doi:10.37044/osf.io/n7qku.
- [10] H. Turki, D. Jemielniak, M. A. H. Taieb, J. E. L. Gayo, M. B. Aouicha, M. Banat, T. Shafee, E. Prud'hommeaux, T. Lubiana, D. Das, D. Mietchen, Using logical constraints to validate statistical information about disease outbreaks in collaborative knowledge graphs: the case of COVID-19 epidemiology in wikidata, *PeerJ Computer Science* 8 (2022) e1085. URL: <https://doi.org/10.7717/peerj-cs.1085>. doi:10.7717/peerj-cs.1085.
- [11] H. Turki, M. A. H. Taieb, M. B. Aouicha, Enhancing filter-based parenthetic abbreviation extraction methods, *Journal of the American Medical Informatics Association* 28 (2021) 668–669. URL: <https://doi.org/10.1093/jamia/ocaa314>. doi:10.1093/jamia/ocaa314.
- [12] O. Pelgrin, L. Galárraga, K. Hose, Towards fully-fledged archiving for RDF datasets, *Semantic Web* 12 (2021) 903–925. doi:10.3233/sw-210434.
- [13] H. Turki, M. A. Hadj Taieb, M. Ben Aouicha, Coupling wikipedia categories with wikidata statements for better semantics, in: *Proceedings of the 2nd Wikidata Workshop*

- (Wikidata@ISWC 2021), 2021, p. 8. URL: <https://ceur-ws.org/Vol-2982/paper-8.pdf>.
- [14] J. E. Labra-Gayo, A. G. Hevia, D. F. Álvarez, A. Ammar, D. Brickley, A. J. G. Gray, E. Prud'hommeaux, D. Slenter, H. Solbrig, S. A. H. Beghaeiraveri, B. Fünfstük, A. Waagmeester, E. Willighagen, L. Ovchinnikova, G. Benjaminsen, R. G. González, L. J. Castro, D. Mietchen, Knowledge graphs and Wikidata subsetting, BioHackathon Europe 2020 (2021). doi:10.37044/osf.io/wu9et.
- [15] V. Nguyen, O. Bodenreider, A. Sheth, Don't like RDF reification?, in: Proceedings of the 23rd international conference on World wide web, ACM, 2014. URL: <https://doi.org/10.1145/2566486.2567973>. doi:10.1145/2566486.2567973.


```

<deceased person> EXTRA p:P31 {
  wdt:P31 [wd:Q5] ;
  p:P570 @<P570_dod> ;
  p:P569 @<P569_dob>? ;
  p:P27 @<P27_citizenship>+ ;
}

<P570_dod> {
  ps:P570 xsd:dateTime ;
  #prov:wasDerivedFrom @<main_reference>+ ;
}

<P569_dob> {
  ps:P569 xsd:dateTime ;
  #prov:wasDerivedFrom @<main_reference>+ ;
}

<P27_citizenship> {
  ps:P27 @<country> ;
  #prov:wasDerivedFrom @<main_reference>+ ;
}

<main_reference> {
  pr:P248 {} ;
  pr:P577 {} ;
  pr:P854 {} ;
  pr:P1476 {} ;
  pr:P6333 {} ;
  pr:P813 {} ;
}

<country> {
  wdt:P31 [wd:Q6256] ;
}

```

Figure 1: The types of statements that can be found in Shape Expressions (ShEx): *Open constraints* (Green), *Indirectly closed constraints* (Yellow), and *Directly closed constraints* (Red) [Source: <https://www.wikidata.org/wiki/EntitySchema:E105>, Input item: *Deceased person* (Brown)].

human (Q497)

No description defined

 edit

[▼ In more languages](#)

[Configure](#)

Language	Label	Description	Also known as
English	human	No description defined	
français	humain	No description defined	

[All entered languages](#)

Statements



EntitySchema for this class	 E3300	 edit
	▼ 0 references	+ add reference
		+ add value

Figure 2: An example for the use of the *EntitySchema for this class* property in the test Wikidata. In this case, there is an EntitySchema associated with the item for "human", which the user finds listed in the record for that concept. The schema itself is accessible through its identifier E3300. (Source: <https://test.wikidata.org/wiki/Q497>).

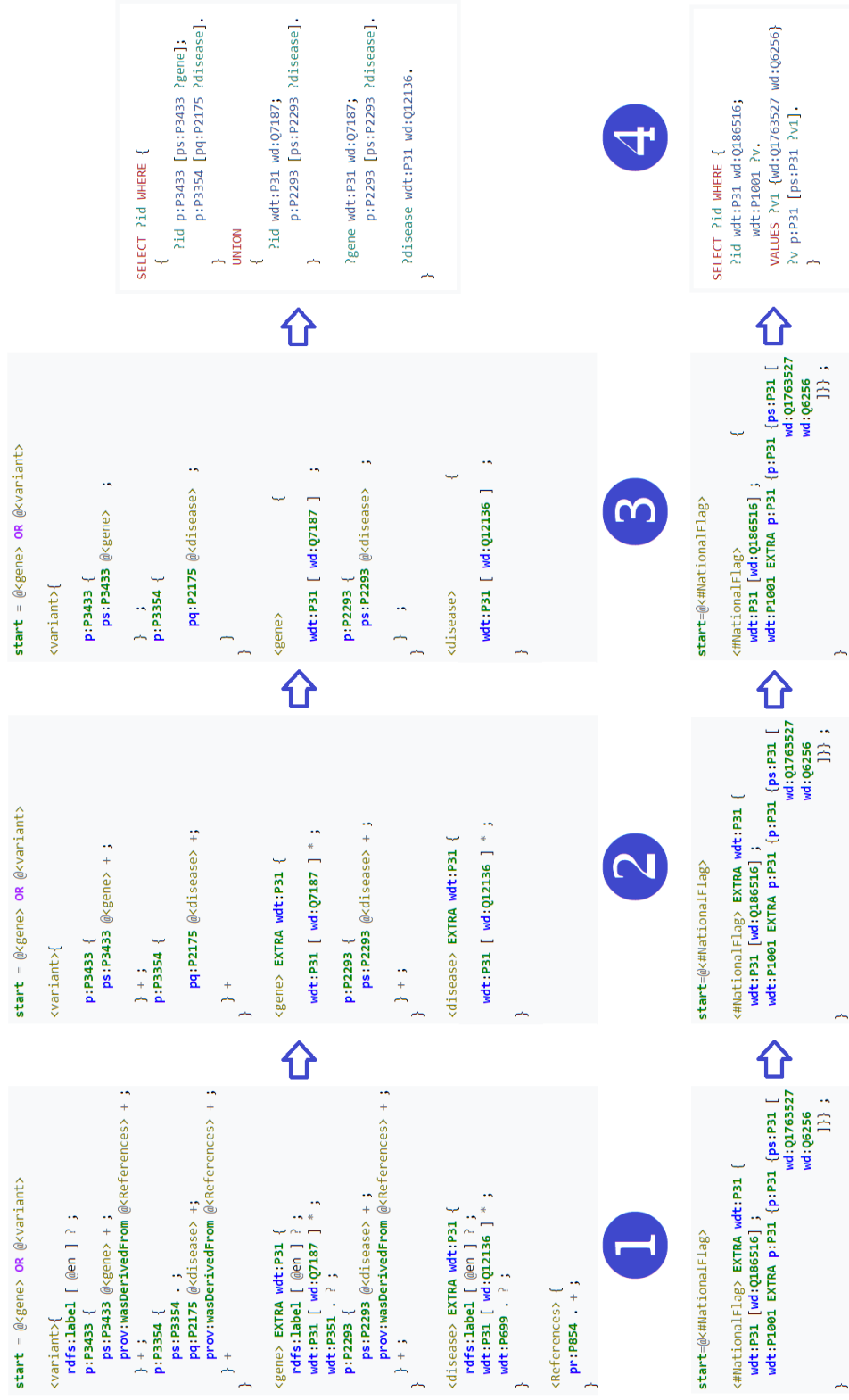


Figure 3: Process for the conversion of ShEx statements into corresponding SPARQL queries: *Extraction of the EntitySchema without comments (1), Elimination of open constraints (2), Elimination of useless non-alphanumerical characters and extra properties (3), and Formulation of SPARQL query (4)* [Source: <https://www.wikidata.org/wiki/EntitySchema:E390> (Gene or Variant with references, top), <https://www.wikidata.org/wiki/EntitySchema:E50> (National Flag, bottom)].

The image shows a Wikidata interface for editing an EntitySchema. On the left, a grey sidebar contains the text "EntitySchema for this class". The main content area displays the following information:

- Entity ID: E19
- Qualifier: occupation, Value: scientist
- Qualifier: country of citizenship, Value: Tunisia
- Reference count: 0 references
- Buttons: + add reference, + add value
- Edit icon: edit

Figure 4: An example of how to use reification to specify how category members are defined in Wikidata. In this example, we add {"occupation", "scientist"} and {"country of citizenship", "Tunisia"} as qualifiers to the *EntitySchema for this class* statement for the *Tunisian scientist* item (Source: <https://w.wiki/7RrE>).

```
1 SELECT ?id WHERE {
2   VALUES ?id {wd:Q80110}
3   ?id wdt:P31 wd:Q186516;
4       wdt:P1001 ?v.
5   VALUES ?v1 {wd:Q1763527 wd:Q6256}
6   ?v p:P31 [ps:P31 ?v1].
7 }
```

Table ?

id

wd:Q80110

Figure 5: An example of the one-line adaptation of a SPARQL query to verify whether a Wikidata item is a solution to it or not. In this case, the input is "Q80110", which is the Wikidata Q identifier for the flag of Canada. The process checks whether this is an *instance of a national flag* [wdt:P31 wd:Q186516], and that it *applies to a jurisdiction* [wdt:P1001], which is a *constituent state* [wd:Q1763527] or *country* [wd:Q6256]. The flag of Canada meets these criteria, and the process verifies this by outputting its Q identifier. (Source: <https://w.wiki/748P>).