PhyQus: Automatic Unit Conversions for Wikidata Physical Quantities

Felipe Vargas-Rojas[1,2], Axel Polleres[3], Llorenç Cabrera-Bosquet[1] and Danai Symeonidou[2]

[1]INRAE-LEPSE, Institut Agro, Montpellier, France
[2]INRAE-MISTEA, Institut Agro, Montpellier, France
[3]Department of Information Systems and Operations Management, Vienna University of Economics and Business, Austria

Abstract

Wikidata is gaining attention to address scientific experimental information. In particular, users can exploit the notions of physical quantities and units of measurement already defined in its knowledge graph. However, when users perform queries over scientific data referring to such data, they can only retrieve the physical quantities in the units of measurement explicitly stored as statements, although the knowledge to transform these quantity values into different units required by the query is already (partially) defined in the units' metadata. We propose PhyQus a query-answering approach that allows to retrieve the physical quantities in any convertible unit by performing unit conversion on the fly based on the query information. To this end, our approach is based in the advanced features of the W3C recommendation SHACL and leverages the ontology of unit of measurements, QUDT. We showcase that the approach is feasible considering two main examples one about cities's area and the other about the boiling point of chemical substances.

Keywords

Wikidata Physical Quantities, Unit Conversions, Numerical Information, SHACL, QUDT

1. Introduction

Wikidata (WD) has become a central and popular repository for not only encyclopedic knowledge, but also as a knowledge base repository for science, e.g., life sciences [1]. Related information includes base facts ranging from properties of locations and places, to properties of genes, as well as physical properties of substances. Many of these properties are described as numerical terms (e.g., area, boiling points, ...) in different standardised units of measurement (km^2, m^2, $^\circ C$, $^\circ F$, etc.).

Yet, while property constraints [2], such as the allowed-units constraint (Q21514353) allow the community to restrict the permitted units of measurement, and while these units are typically inter-convertible, Wikidata's query service only provides answers in the explicitly stored units. Limited support for value conversion, exists in the sense that particular units such as for instance Q712226 (square kilometre), provide explicit conversion factors, via property P2442 (conversion to standard unit), or PP2370 (conversion to SI unit), but this information is typically incomplete

Wikidata'23: Wikidata workshop at ISWC 2023
luis-felipe.vargas-rojas@inrae.fr (F. Vargas-Rojas)

within WD and does not cover more complex conversions, such as for instance $^{\circ}C$ to $^{\circ}F$, which involve more complex equational knowledge than multiplication factors.

The previous issues could be prevented by precomputing all possible unit conversions, which would allow the query service to provide all the complete results. However, this is currently unfeasible, regarding that the latest version of WD contains 158 properties of quantity values and they can be applied to millions of entities. To illustrate, consider the "area" property, which is used from around 5,813,704 entities [1], and according to the "allowed unit" constraint, can be expressed in 20 different units [2]. Consequently, a complete materialisation would demand over 116,274,080 additional triples.

It seems obvious, that leveraging existing work in the Semantic Web field on defining exhaustive ontologies for units of measurements, such as QUDT [3] could therefore add great benefits in the usability of data within WD. To this end, in the present paper we describe our approach, PhyQus, to convert physical quantities within WD automatically and on the fly, leveraging the entire - and extensible - knowledge in ontologies such as QUDT, in a manner transparent to the end user of Wikidata's query interface.

In the remainder of this paper, after discussing related works Section 2, we describe our approach in Section 3, illustrated by examples, showcasing automated query-answering in different target units. We discuss potential impact, in terms of the incompleteness/heterogeneity of current unit data within Wikidata as well as an empirical evaluation in Section 4, before we conclude in Section 5.

2. Related Work

The problem of performing unit conversions involves certain requirements and steps. In Section 2.1 we explain the most common approaches. Besides, in Section 2.2 we present alternatives to express data derived from mathematical computations that enrich the initial data graph. Finally, in Section 2.3 we present three relevant unit ontologies considering that their annotations and data models are required to perform this task.

2.1. Current Unit Conversion Approaches

The unit conversion task relies on a notion of quantity values. Different models can represent this notion, for instance the ontology QUDT and Wikibase (used in the Wikidata) propose properties and classes for that goal. A *quantity value* is a pair $(value, unit)$ which expresses the numerical *value* of a quantity with respect to a *unit of measurement* [3].

Bearing in mind the notion of quantity, the most traditional way to perform this task is by defining complex SPARQL queries. This approach was followed by [4], the user should add manually in the query the conversion functions and if required to construct the new quantity value. Another alternative is to export the numerical information along with the unit of measurement identifier, therefore to use programming language libraries such as Java [3] or

[1] https://w.wiki/7qta
[2] https://w.wiki/7qth
[3] https://github.com/unitsofmeasurement/unit-api

Python[4]. Users should define manually the mappings between the library and the unit ontology and expect that the library supports the units they are interested in.

2.2. Leveraging conversion rules and equations automatically

Property equations [5] were an early proposal to automate numerical conversions in RDF data, leveraging equations in terms of automated query rewriting, relying on simple linear equations, expressed as strings, that could be automatically rewritten in terms of SPARQL BIND expressions. This approach has been later extended to cover also more complex conversion rules, QBrules [6], including aggregation over dimensions in DataCubes. QAVAN [7] presents other alternative to produce derived data on the fly during query-answering. This approach allows to define mathematical formulas in RDF along with the data. Besides, it permits more advanced calculations (e.g., power, root squares, sine, cosine, and so on..). QAVAN introduces the concept of Actionable Numerical Relationship (ANR) as a kind of rule that contains a mathematical formula in their body. QAVAN implements the ANRs based on SHACL ValueRules. Thus, ANRs enable to produce new triples derived from numerical calculations. In constrast to QAVAN, where the ANRs are stored in advance, our approach PhyQus requires to generate them "on the fly" since the ANRs rely on parameters given by the query (e.g., the desired unit of measurement) that are not known in advance.

2.3. Unit Ontologies

We explore three of the most used unit ontologies: (i) UO, the unit ontology from the OBO foundation [8]; (ii) OM, Ontology of units of measure and related concepts [9]; and (iii) QUDT, quantities, units, dimensions and data types ontologies [3]. Although these three ontologies are a rich resource of annotations about units, not all of them aim to facilitate the unit conversion task. For instance, UO lacks annotations about conversion factors whereas OM requires a programming language to express functions with recursion in order to perform the unit conversions. Conversely, QUDT unit conversions can be computed in a W3C standard such as SPARQL without requiring external third-party software. Therefore, our approach is based in this technology.

3. Approach

In this section we describe our approach PhyQus. For this approach we assume the data conforms to the model of quantities used in Wikidata. This approach is inspired in the notions addressed in QAVAN [7] and use the same concept of Actionable Numerical Relationship (ANR) to refer to our production rules. In PhyQus the ANRs are ephemeral and instantiated considering the information of a given query. PhyQus relies on QUDT to obtain the conversion factors and offsets, as well as in SHACL Advanced Features (SHACL-AF) for two main processes: (i) perform unit conversions and (ii) add new derived physical quantities to the data graph through the definition of ephemeral ANRs. More precisely, we incorporate new SHACL functions to

[4]https://github.com/hgrecco/pint

enable unit conversions, and new SHACL plugins to enable "on the fly" construction of new quantity values.

In this work we use some SHACL [10] notions, in SHACL a shape represents a group of constraints associated to a set of target nodes. The constraints are evaluated against each node to conclude if the data is valid against that SHACL shapes. During the evaluation a node is refereed as the *focus node*. For instance, a target can be the class City whilst a focus node can be the city of Montpellier. The same concepts are used to the definition of ANRs.

3.1. Performing Unit Conversions and Create Wikidata Quantities

Funtion	Description	Domain	Range
toTargetUnit	Transform a QuantityValue to a given target unit and returns the numerical value	xsd:Double qudt:Unit (source) qudt:Unit (target)	xsd:Double
toWDfromQUDT	Transform a WIKIDATA unit into a QUDT unit	wikibase:Unit	qudt:Unit

Plugin	Name	Description
SHACL rule **SHACL expression**	WDQuantityRule createWDQuantity	Enable WD quantity values as outputs Take a Wikidata property, a numerical value, and a Wikidata unit and constructs a Wikidata quantity value instance.

Table 1
SHACL functions and plugins to enable unit conversions and produce quantities conforming the Wikidata model.

Table 1 presents the extensions added to SHACL-AF. We add two SHACL functions: *toTargetUnit* to transform a value into a target unit and *toWDfromQUDT* to transform a unit given in the Wikidata model to QUDT. These functions take advantage of the mappings to QUDT existing in Wikidata through the direct link property wdtn:P2968. Besides, Table 1 also presents two SHACL plugins, whilst SHACL functions are implemented as SPARQL queries and can be added through RDF, the plugins require to add some lines of code in the SHACL-AF implementation. These two kind of extensions are considered and are straightforward to add in SHACL-AF. In the next section we explain how to align these extensions with the query-answering process.

3.2. Query-Answering Process

The query-answering steps are described in Figure 1 along with a simple example considering the automatic unit conversion for data related to the city of Montpellier. The approach takes as input a query Q retrieving some physical quantities of WD. In the example we mention the property p:P2046, which refers to the area of an object, however the query can be about others properties (e.g., the boiling point of a substance). Note that the query requires this property in the unit square miles (wd:Q232291). Shapes having as target, subjects of the Wikidata physical quantities, should be added in order to activate the automatic unit conversion. The example

shows a shape called AreaShape and having as target subjects of the property area. Last input is a data graph, in this example is a graph containing information about the area of the city Montpellier (wd:Q6441). This graph contains the Montpellier's area only in square kilometres (wd:Q712226). The initial query without our approach is going to return an empty result set. The steps to enable query-answering with automatic unit conversion are described as follow:

1. Query Interpretation. Taking into account the shape's target, in this step PhyQus identifies the properties involved in the query that have also an associated shape. In the given example, it is the property area. From the triple patterns in the query, PhyQus obtains the variable that represents the targets (e.g., ?city). Besides, a map is generated from the query, which keys are the properties and which values are a list of required units. The example shows only the unit square mile, however, more units can be required in the same query.
2. ANRs Generation. PhyQus iterates on the map producing one ANR for each pair (property, unit). The new ANRs that are added to the shape using the property sh:rule are ephemeral and only exist during the query evaluation. These new ANRs, use the functions and plugins defined in Section 3.1. Given the space constraints, the complete ANRs are only shared in the experiment repository. We also add a policy to avoid the proliferation of calculations. If the current focus node (e.g., a city) already has a quantity value in the required unit, then we avoid to compute a new one. We express this policy as a SHACL constraint taking advantage that our approach is based on SHACL-AF. This policy requires further discussion since there are different reasons for which a focus node have several times the same property: (i) erroneous data, (ii) data in different time point, (iii) the same value in other unit of measurement but equivalent.
3. Update Target Nodes. First, PhyQus removes from the query all the statements about the identified properties. After that, for each identified property it adds a triple pattern having as predicate this property and as object a blank node. This pattern assures that at least the focus node has once the property, which can therefore enable the unit conversion.
4. Perform ANRs. Considering the ephemeral ANRs and the target node query the ANRs are performed in order to produce a virtual graph containing the new quantity properties. In our example, the derived graph contains triples about a new area property for the city of Montpellier with the quantity value in square miles. Note that this information was not presented in the data previously.
5. Query Evaluation. The query is evaluated against the union of the new virtual graph and the initial graph. Therefore, it produces a result set. In the given example, a result with the Montpellier's area in square miles.

4. Evaluations and Results

PhyQus Evaluation is illustrated with two Wikidata examples: one about the area of cities and their subclasses, and other about the boiling point of different substances. The former evaluation is carried out to demonstrate the limits of Wikidata units of measurements and to stress the necessity of using unit ontologies such as QUDT. Experiments are shared in this

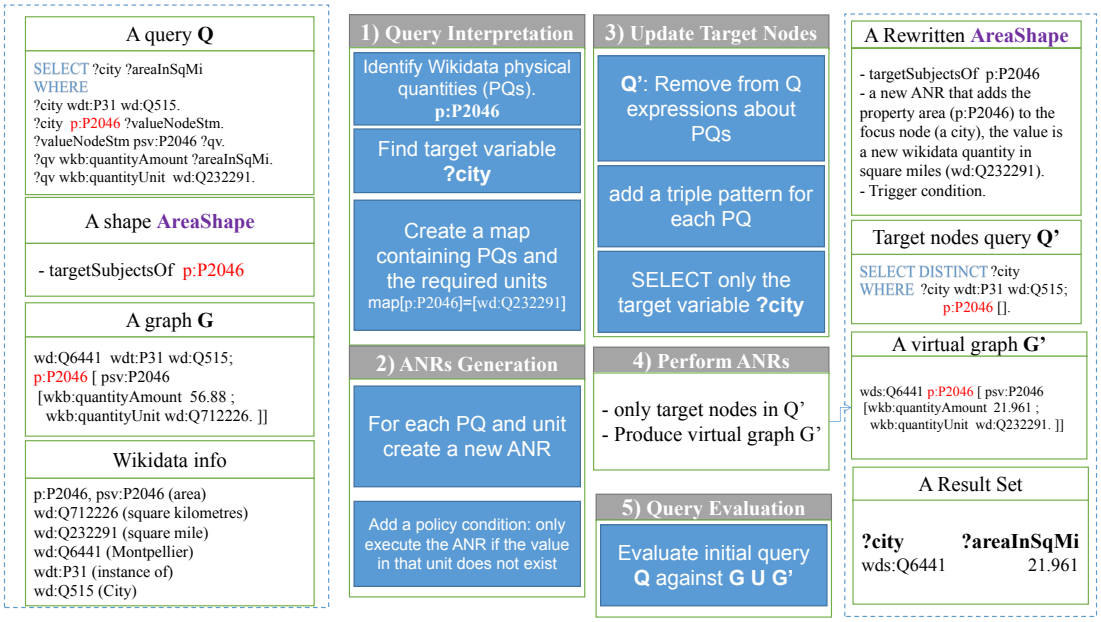

Figure 1: PhyQus detailed steps. Inputs are: a query, different shapes using the *targetSubjectOf* property and a data graph. PhyQus produces a rewritten shape with ephemeral ANRs, a temporal graph and a result set with the query results.

repository [5].

4.1. Evaluation of Cities's Area

We evaluate PhyQus taking into account two main scenarios: (i) perform a query using the traditional Wikidata SPARQL endpoint, and (ii) evaluating the same query with PhyQus.

Table 2 shows the evaluation of different queries about the area of a given city in different units. Each "Unit Label" in the table represents a query that retrieves that specific unit. These results demonstrate that PhyQus improves the number of query results. For instance, the query about "square mile" only returns about 4% of the results for the Wikidata endpoint whereas the same query evaluated with PhyQus is able to retrieve about 103% of the data. We explore the data to find why it is returning more than the 100% of the data despite our policies. We found, for instance that a city such as Bönnigheim (wd:Q61837) has six times the property area in the same unit. If the unit is not the required by the query, PhyQus will return six new quantity values in the same required unit. That is because PhyQus, in the current version, cannot access the produced physical quantities and only perform the validation in the original data.

We observe as well that for the query about "square km" PhyQus shows less results than the Wikidata endpoint. It can be given to our preprocessing where we transform the data types to double and not all the values are correct numerical values. We remark that, although PhyQus

[5] https://github.com/felipe-vargas-inrae/phyqus/

presents better results, it is also relevant to perform further improvements in order to handle the data incompleteness, heterogeneity and inconsistency thus contributing to the data quality.

Approach	Unit Label	Unit Wikidata	# of results	# of cities	% (x/20640)
WDEndpoint	square mile	wd:Q232291	802	20640	3,885658915
WDEndpoint	square meters	wd:Q25343	923	20640	4,471899225
WDEndpoint	square km	wd:Q712226	39662	20640	192,1608527
PHYQUS	square mile	wd:Q232291	21316	20640	103,2751938
PHYQUS	square meters	wd:Q25343	20971	20640	101,6036822
PHYQUS	square km	wd:Q712226	34489	20640	167,0978682

Table 2
Comparing Wikidata Enpoint vs PhyQus for queries about the area of a city in different units.

Approach	Unit Label	Unit Wikidata	# of results	# of substances	% (x/1053)
WDEndpoint	kelvin	wd:Q11579	130	1053	12,34567901
WDEndpoint	celsius	wd:Q25267	450	1053	42,73504274
WDEndpoint	fahrenheit	wd:Q42289	472	1053	44,82431149
PHYQUS	kelvin	wd:Q11579	959	1053	91,07312441
PHYQUS	celsius	wd:Q25267	1025	1053	97,34093067
PHYQUS	fahrenheit	wd:Q42289	1014	1053	96,2962963

Table 3
Comparing Wikidata Enpoint vs PhyQus for queries about the boiling point of substances in different temperature units.

4.2. Evaluation of Substance's Boiling Point

This second evaluation shows that PhyQus can handle more complex unit conversion such as those about temperature units, which require not only a conversion factor but also an offset. Note that the Wikidata model does not allow that kind of conversions. Results in Table 3 exhibits a better number of results for the queries that were evaluated in PhyQus covering about 90% of the substances. The results for PhyQus about different units were not uniform because of the issues of data quality already mentioned on the previous evaluation.

5. Conclusions

In this work we present PhyQus, an approach to automate unit conversion during query evaluation. We exploit SHACL-AF and leverage the unit ontology QUDT for that goal. PhyQus produces on the fly ephemeral Actionable Numerical Relationships (ANR) relevant to answer a given query. We also remark that, although the Wikidata model offers some properties to perform unit conversions, these annotation are limited and do not allow unit conversion for cases such as temperature units $^{\circ}C \rightarrow {}^{\circ}F$. This scenario stress the idea of using a more robust unit ontologies such as QUDT. We evaluate PhyQus against two scenarios: cities's area and boiling points of substances. Our results demonstrated that PhyQus is able to exploit the numerical

knowledge about conversions to provide more results under certain queries that the traditional Wikidata endpoint could not handle. For instance, a query retrieving the area of a city in square miles returns only 802 results whereas with PhyQus it provides 21316. **As future work**, we plan to conduct further evaluations. Besides, the approach is also suitable to repair violations of constraints about units of measurement. There is also an interest to express mathematical formulas in different experimental domains which inputs and outputs are quantity values.

Acknowledgments

This work was supported in part by INRAE, #DigitAg, by the French National Research Agency under the Investments for the Future Program, referred as ANR-16-CONV-0004 and by the the EU project STARGATE H2020 952339 (https://stargate-hub.eu/).

References

[1] A. Waagmeester, G. Stupp, S. Burgstaller-Muehlbacher, B. M. Good, M. Griffith, K. Hanspers, O. L. Griffith, H. Hermjakob, T. S. Hudson, K. Hybiske, S. M. Keating, M. Manske, M. Mayers, D. Mietchen, E. Mitraka, A. R. Pico, T. Putman, A. Riutta, N. Queralt-Rosinach, L. M. Schriml, T. Shafee, D. Slenter, R. Stephan, G. Tsueng, K. Thornton, R. Tu, S. Ul-Hasan, E. Willighagen, C. Wu, A. I. Su, Science forum: Wikidata as a knowledge graph for the life sciences, eLife Sciences 9 (2020). URL: http://eprints.cs.univie.ac.at/6703/. doi:10.7554/eLife.52614.

[2] N. Ferranti, A. Polleres, J. F. D. Souza, S. Ahmetaj, Formalizing property constraints in Wikidata, in: Proceedings of the 3rd Wikidata Workshop (co-located with ISWC2022), 2022. URL: http://polleres.net/publications/ferr-etal-2022WD.pdf.

[3] R. Hodgson, P. J. Keller, J. Hodges, J. Spivak, QUDT-quantities, units, dimensions and data types ontologies, USA Available http://qudt. org March 156 (2014).

[4] F. Martín-Recuerda, D. Walther, S. Eisinger, G. Moore, P. Andersen, P.-O. Opdahl, L. Hella, Revisiting Ontologies of Units of Measure for Harmonising Quantity Values–A Use Case, in: International Semantic Web Conference, Springer, 2020, pp. 551–567.

[5] S. Bischof, A. Polleres, RDFS with attribute equations via SPARQL rewriting, in: P. Cimiano, O. Corcho, V. Presutti, L. Hollink, S. Rudolph (Eds.), The Semantic Web: Semantics and Big Data – Proceedings of the 10th ESWC (ESWC2013), volume 7882, Springer, Montpellier, France, 2013, pp. 335–350. URL: http://www.polleres.net/publications/bisc-etal-2013ESWC.pdf.

[6] S. Bischof, A. Harth, B. Kämpgen, A. Polleres, P. Schneider, Enriching integrated statistical open city data by combining equational knowledge and missing value imputation 48 (2018) 22–47. URL: http://polleres.net/publications/bisc-etal-2017JWS.pdf. doi:10.1016/j.websem.2017.09.003.

[7] F. Vargas-Rojas, L. Cabrera-Bosquet, D. Symeonidou, QAVAN: Query-answering approach for actionable numerical relationships over knowledge graphs, 2023. Manuscript submitted for publication.

[8] G. V. Gkoutos, P. N. Schofield, R. Hoehndorf, The units ontology: a tool for integrating units of measurement in science, Database 2012 (2012) bas033.

[9] H. Rijgersberg, M. Van Assem, J. Top, Ontology of units of measure and related concepts, Semantic Web 4 (2013) 3–13.
[10] D. Kontokostas, H. Knublauch, Shapes Constraint Language (SHACL), W3C Recommendation, W3C, 2017. Https://www.w3.org/TR/2017/REC-shacl-20170720/.