Data Representation for Social Data Science and Humanities with Django and Wikibase: the *PhytoLex* Database Case

Julia Amatuni, Ivan Bibilov, Kira Kovalenko, Olga Tushkanova, Ivan Usalko
European University at St. Petersburg
St. Petersburg, Russia
{iamatuni, ibibilov, kkovalenko, otushkanova, iusalko}@eu.spb.ru

***Abstract*—The article discusses the main challenges in data storage, organisation and representation for humanities and social science projects. Based on the case of a historical Russian plant names database, the authors offer applied solutions for such projects, including an originally developed adapter for transitioning data from Django to Wikibase. The adapter allows utilising the advantages of the Django framework and Wikidata infrastructure, preventing the data format from becoming obsolete and making queries more flexible with SPARQL.**

I. Introduction

Representation of data and knowledge in an accessible and advantageous way is one of the significant tasks in the 21st century for industry and academia. In the case of social science and humanities, general software-related problems are often accompanied by notably less funding, a shortage of consistent infrastructural support and weakness of interdisciplinary dialogue.

We confronted these issues in our project *PhytoLex* – a database of historical plant names in the Russian language. We believe our findings to be of value in similar projects searching for practical solutions.

Hence the purpose of this paper is twofold. Firstly, we propose to discuss the central issues in constructing modern digital knowledge, particularly in social data science and humanities projects. Secondly, based on our findings and the development of *PhytoLex*, we will offer applied tools for the presented challenges.

II. Critical Challenges in Building a Modern System of Digital Knowledge for Social Science and Humanities

A. Pluralism of storage formats

In the pre-digital era, libraries and archives were primary structures for storing knowledge in various paper forms. Transition to the digital age placed a technical device and software as an intermediary between a knowledge base and a consumer.

Nowadays, there are numerous relational and NoSQL databases on the market that have their own unique internal storage formats. There are also examples of large databases with their exclusive internal structure.

While it is possible to save data as an archive in general formats of SQL commands, RDF, JSON or others, the pluralism of internal formats may lead to two difficulties. The first is compatibility since these formats, as well as archives themselves, evolve into different versions over time. The second is the possibility of a format's withdrawal from the market when a company stops its development and support.

B. Representation of the Data

The vast knowledge humanity inherited on paper exists in a variety of forms, from written and printed books to news articles and academic papers. These data representation forms have been evolving with the standards of corresponding professional areas. The structure of an academic paper provides a good example. Keywords, an abstract, introduction and conclusion allow a reader to glance through the research presented and compose a quick first impression. Thematically divided chapters enable one to address the most relevant subjects depending on one's interests.

However, the most common digital data representation standards often lack convenience for an ordinary researcher, particularly in social science and humanities.

Many private and public archives digitalise their data by scanning and publishing the documents in PDF formats. This process is crucial for building a digital system of knowledge. It is expensive and demanding in terms of funding and human effort. However, it is arguably only the first step on a long way toward making the data of these documents accessible. Even if the text of a published PDF is recognised and a search by keywords is available, finding relative information can still be challenging and will involve a manual read of all the documents.

Another commonly used format of data representation is Excel or CSV sheet. It allows processing the data with modern advanced technology, but it demands an advanced level of skill in programming and a professional understanding of machine data preprocessing.

C. Data Structures, efficient search and the importance of collaboration

Two main factors define the structure of a future database: particular characteristics of the data itself and the types of planned data manipulation.

The same factors dictate the type of search most efficient for particular data. For this reason, there are various modern search tools on the market, but there is no universal solution suitable for all data sets.

Therefore, the development of a convenient data structure and an efficient search demand expertise both technical and in the subject area. This demand is best met in interdisciplinary and well-managed collaborative work of different specialists, which is expensive. Most commonly, specialists in social science and humanities delegate the creation of a database to outsourced commercial companies not specialising in scientific projects. The development in these instances is often based on an initial technical specification rather than an agile methodology. As a result, the created base formally contains all the necessary data, but it does not provide a convenient search mindful of all the particular qualities of the data.

D. Software obsolescence

Another significant challenge in constantly changing digital products is obsolescence. The requirements for functionality, usability, aesthetics and ethics are shifting over time. The ideal software product or a database of the 1960s, while relevant for our understanding of the history of technological development, often lacks practical value.

E. Limited funding and shortage of institutional support

The digitalisation of knowledge includes several stages, mainly the collection and preprocessing of the data, the development of the database structure with efficient search tools and a website, and finally, the support of the created system.

With often limited funding granted to research in social science and humanities, developing and maintaining such projects following modern technologies becomes particularly challenging without the institutional backing of research centres or universities. Thus reducing the costs of software development and support may prove critical for the success of such research projects [1].

In our project, we address these challenges by transforming the *PhytoLex* database into the Wikidata format using Wikibase software.

III. Wikibase: Pros and Cons

For the development of PhytoLex, we considered several possible solutions.

For example, one of the open-source software accessible for such projects is OpenLink Virtuoso, a free edition of the Virtuoso project – a combined multi-model DBMS, Knowledge Graph, and Data Virtualisation platform [2]. Apart from other advantages [3] and disadvantages [4] of this platform, we consider Virtuoso's lack of broad community support a crucial negative factor in preventing the data format from becoming obsolete.

In comparison, Wikibase, as the open-source operating Wikidata, is one of the largest and most popular knowledge graphs on the modern Internet [5]. Hence Wikibase has certain advantages regarding the challenges described in the previous chapter.

A. Open source software: availability and worlds community support

As we already mentioned, Wikibase is open-source software. Even storage of the data in Wikibase.cloud is free if the data is open to everyone. All of it makes Wikibase easily accessible for projects with limited funding. We see more examples of Wikibase projects over the last few years, particularly in social science and humanities [1], [6], [7].

Wikibase offers a promising solution to the problems of obsolescence and the support of software. Since it is a core part of the globally-used Wikidata, it has a well-designed roadmap [8] and development strategy [9]. It also receives funding from major players in the IT industry. Therefore, as K. Thornton and her colleagues note in the case of their Wikibase project: «From the perspective of digital preservation, donating the WeChangEd dataset to Wikidata will result in greater longevity of the data» [10].

There could be another advantage of such software relevant to some projects. It accommodates an infrastructure for the co-working of a large number of participants. This, as revealed in the case of the Italian Library Association initiative, might not only stimulate the development of relevant tools but also enhance the interest and involvement of the broader public in substantial socio-cultural projects [11].

B. Out-of-box services and a wide range of tools

The core infrastructure of the Wikibase consists of several elements: MediaWiki (the software of Wikipedia), Wikibase Repository and Wikibase Client, MariaDB relational database, Elastic Search, Blazegraph and SPARQL framework. All the elements are conveniently integrated into a Docker container for easier set-up. The broader infrastructure around Wikibase includes a variety of bots and tools for data management.

The standard Wikidata components allow a relatively smooth integration of knowledge from a local database into the Wikidata graph (for instance, the open-source WDI tool [12]). The integration of a particular project into a highly advanced ecosystem of digital knowledge thus becomes more accessible.

This broad variety of services and tools available with Wikibase software is designed to meet the most common and standard demands for data management. However, it is evident from examples of Dennis Diefenbach's thorough research [5], the PhytoLex project, and other cases [6], that the more complex tasks might still require further development.

Wikibase data and knowledge storage format is a set of SQL tables [13], so the power of already installed DBMS reduces the cost of maintaining the knowledge base. Indexing and data search are implemented as external relational storage services in RDF triplet storage [14].

In the following chapters, we will demonstrate the challenges and solutions our team provided for the *PhytoLex* database.

IV. PhytoLex database

A. PhytoLex as a Database of the Russian plant names of the 11th-17th c.

The *PhytoLex* is a relational database of the historical plant names recorded in the Russian language in the 11th-17th c. [15], originally modelled in the *Django* framework [16].

Before the development of the *PhytoLex*, the project participants stored the information in various forms, namely notes written in books or on paper, in Word or Excel files. This information contained data relevant to the professional interests of a particular researcher: it could be plant names, citations, information about sources, etc. For the database, these pieces of information had to be unified, structured and distributed to classes.

Plants and plant names were to be examined from a predominantly linguistic point of view, hence the three classes of the database (*Lexeme*, *Etymology* and *Usage*) contained essentially linguistic data: words denoted plants and their phonetic and morphological variants, the etymology of plant names, root words, source language in case of borrowing, citations, etc. In the *Original*, *Source*, and *Publications* classes, manuscripts, books, articles, and dictionaries were described as sources for the citations. *Plants* class contained information about species according to modern scientific knowledge: their life forms, type (wild/domestic), and common and Latin names based on the botanical classification [17]. Two classes included data about places where plants were grown, collected or sold and organisations (libraries, institutions and so on) to specify manuscripts' location.

In order to reduce misspelling mistakes, the core fields of PhytoLex were provided with drop-downs controlled by corresponding vocabularies. These fields are *Plant parts*, *Functions*, *Dosage form in medicine*, *Places*, *Organisations*, *Source language of the lexeme*, *Translation methods* of plant names from foreign sources, *Language of foreign equivalents, Publication Types,* and *Literature genres*.

The scheme of the PhytoLex database for the Russian plant names recorded in the 11th-17th centuries is presented in Fig. 1.

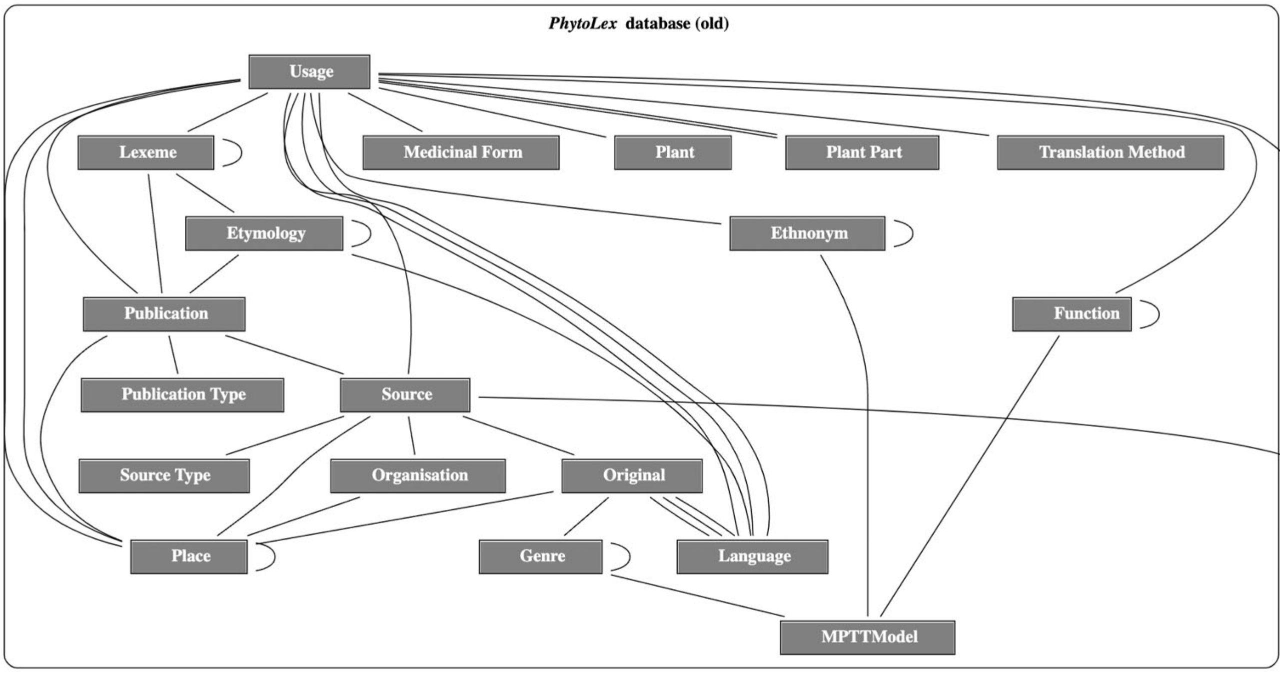

Fig. 1. Data model of the *PhytoLex* database (Russian plant names of the 11th-17th c.)

By the end of the project, the database included more than 15,000 records and was available online [18]. Using an in-built query set, one could find plant names from a particular period, to observe the lexical variants, to determine the meaning of the plant names which were not given in the historical dictionaries. Based on the *PhytoLex*, a number of researches were carried out [19]. Some materials from *PhytoLex* were also used for the *Dictionary of the Russian language of the 11th-17th c.* [20].

B. PhytoLex transformation: anthropological and time shift

A few years later, the PhytoLex team resumed its work but this time with the institutional support of the European University at St. Petersburg. The interdisciplinary team developing the project in the Centre for Machine Learning, Data Analysis and Statistics (MAST) [21] includes specialists in linguistics, history, anthropology, software engineering and project management. This involvement of different professionals allowed us to reconstruct the database in accordance with the data characteristics and the requirements for its manipulation.

The new project focused on plants on the Russian Empire territory in the early modern period — both the use of plants and the dissemination of empirical and expert knowledge. It reveals the role of plants in various spheres of human activity (consumption as food, medical, decoration, construction usage, ritual practice, etc.) and the distribution of these functions among social classes.

The cross-disciplinary collaboration made it possible to look at the research problem from various perspectives, which resulted in several modifications in the *PhytoLex*. We created the new classes *Citations*, *Human* and *Animal diseases* and *Pictures* (photocopies of herbaria and book illustrations). We also considerably broadened the list of functions, as a lot of treatises describing plants' use in various spheres of human life were published during the 18th c.

It became possible to upload PDF files and highlight fragments containing plant names. Now the text from the text layer of the highlighted fragment can be automatically included in the relevant field of the database in the adjacent part of the screen (Fig. 2). For text recognition from PDF files, a model for OCR in publications of the 18th century was developed [22].

With the increasing complexity of the database (Fig. 3), the restrictions of its functionality started to be more visible. The in-built search possibilities were not satisfactory enough for the first project, and now it was even harder to meet all the requirements for data retrieval. Also, the database did not supply output forms, as it was not possible to download selected data.

In search of a solution to these issues, our team decided to test the Wikibase software.

Fig. 2. Component for PDF mark up

V. Migration to Wikibase

A. The WikibasIC container

WikibasIC information container is one of the standard technologies of Wikibase we used for synchronisation of implemented data. The WikibasIC container uses Wikidata components [23] and provides a high level of compatibility of formats (both storage and presentation of knowledge). It is a set of services ready for set-up and consists of Wikidata components. The container distribution package is updated synchronously with the components update. To preserve the limited resources we had for the PhytoLex project, we optimised the work of the container to 4GB.

B. The WikibasIC container

As we previously mentioned, the problem of search accessibility is one of the most crucial challenges in the field of digital knowledge. Wikibase uses the index in the form of an RDF store and provides SPARQL as the language for the search [24].

SPARQL [25] was developed by the W3C to enable deep searches on the World Wide Web. While this approach presents some disadvantages [26], it proves to be effective for depth-first search across a vast subnet of nodes (sites), on the condition the automatisation of ontological classification is provided [27].

In addition to depth-first search, SPARQL technology allows easy out-of-the-box access to all available information of the particular Wikibase. This feature is implemented with a SPARQL point and can be supplemented by opening access to the SPARQL query builder and editing services ([28], [29]).

C. The Django adapter

When transferring the knowledge to Wikidata and testing several plugins developed for Mediawiki ([30], [31], [32]), we faced limitations of standard MediaWiki forms, which are:

1. Forms are described by a markup language that requires special knowledge.
2. It is impossible to implement the logic of the form's related fields.

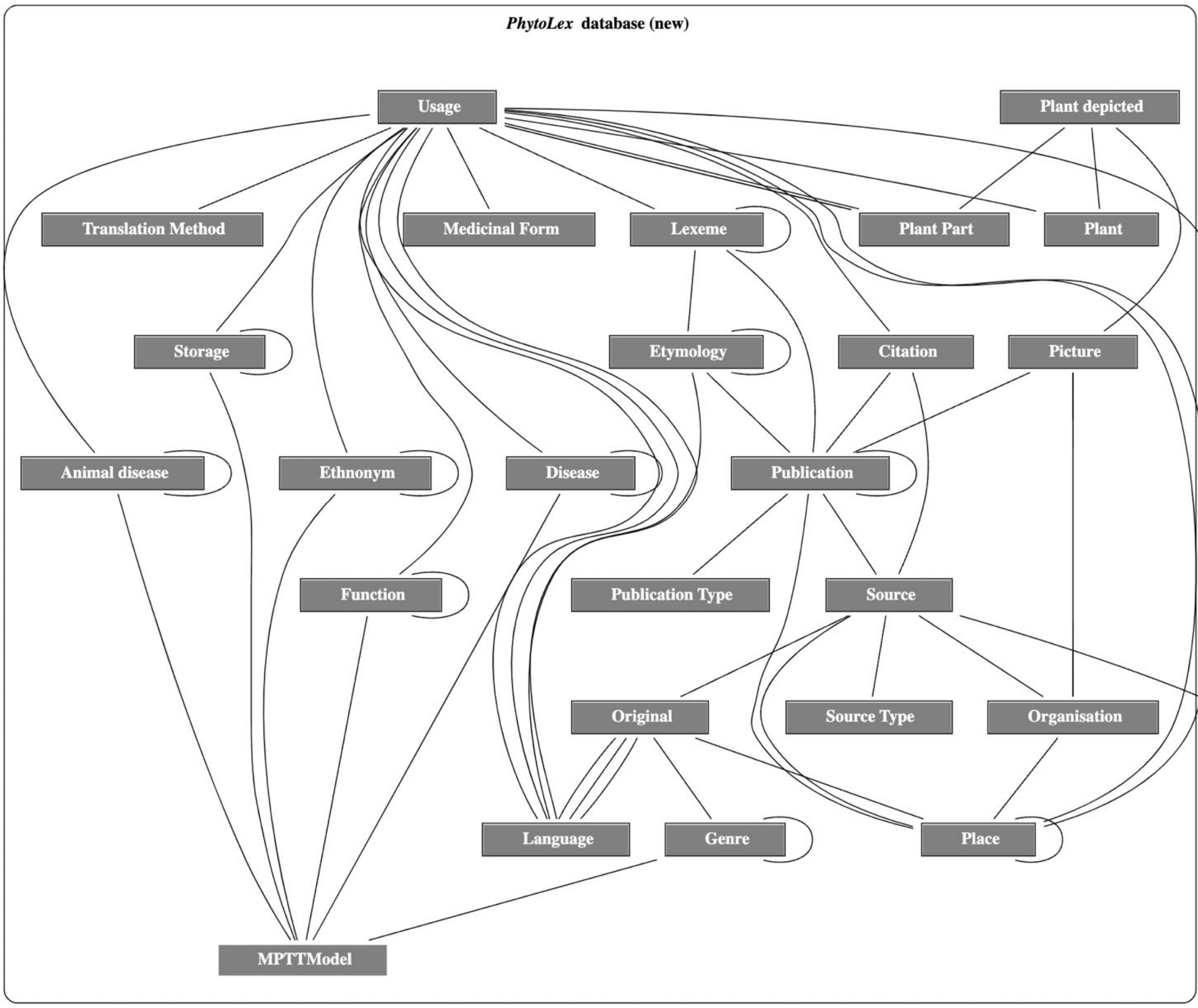

Fig. 3. Data model of the *PhytoLex* database (Russian plant names of the 18th c.)

3.Design of forms tied to Mediawiki styles.

4.Impossible to implement modal forms.

5.Impossible to implement complex flows such as wizards.

6.There is no option for role-based access to forms.

To address these issues, we developed an adapter for the integration of the *Django* framework, that allowed us to create predefined forms (admin domain) and specialised forms for data entry. The Django-Wikibase adapter is an integration layer between Django and Wikibase that works at the database dialect layer for Django [33] (see Fig. 4). It allows the implementation of flexible solutions for synchronisation between an existing database and its open-data copy in Wikibase.

The architecture of the Django-Wikibase adapter is built on a query compiler and an action-API adapter (Fig. 5). The database query flow in Django presents a sufficient level of abstraction, allowing the implementation of the Wikibase query function (the compiler must be in SPARQL).

Due to the extreme simplicity and flexibility of the Wikibase model ([34]), Django concept mapping can be implemented in many ways. At the moment, we are using a direct way, as shown in Fig. 6.

However, there are still some difficulties in implementing the Django-Wikibase adapter. The most significant is the delayed one-way synchronisation between the SQL store (MediaWiki) and the SPARQL service (wdqs) and the lack of Wikibase-type validation.

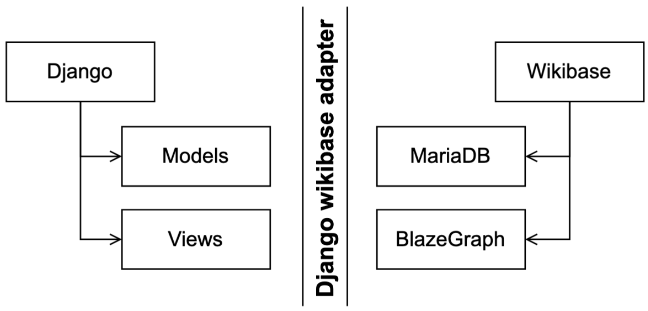

Fig. 4. Django-Wikibase adapter

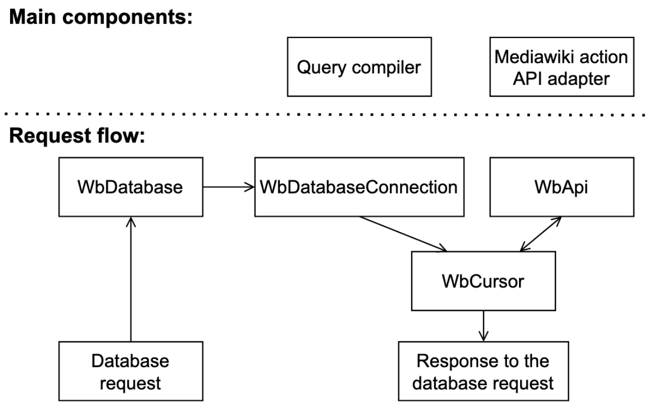

Fig. 5. Architecture of the Django-Wikibase adapter

To solve the first problem, we built the code into the adapter for synchronised data updates in SQL and SPARQL stores. Standard MediaWiki action API allows us to work with SQL storage. All the data sample requests go through a SPARQL point provided by the wdqs service. The lack of Wikibase type validation we compensate for with special checks transferred to the adapter module.

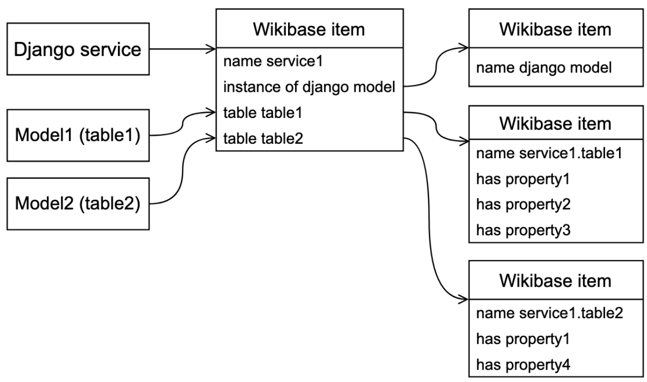

Fig. 6. Django concept mapping

VI. Conclusion

This paper discusses the challenges the digitalisation of knowledge presents with a particular focus on the areas of social science and humanities. We address such crucial issues as storage format pluralism, software obsolescence, efficient search, limited funding and shortage of institutional support.

To resolve these challenges, we tested Wikibase. It is open-source software for WikiData, supported widely by a global community and thus offering promising opportunities for scientific and cultural projects. Most importantly, Wikibase solves some of the mentioned problems, namely software obsolescence, smooth integration into the web of world digital knowledge and conservation of resources.

We implemented the Wikibase software in the *PhytoLex* project.

The *PhytoLex* was originally a relational database modelled in the Django framework. It contains the historical plant names recorded in the Russian language in the 11^{th}-17^{th} c. The existing model of the *PhytoLex* did not meet the requirements the research team had for search and data management. Whereas Wikibase format offered such benefits as general technical support, the flexibility of the data mining with SPARQL and the option to save search results in various formats.

When transferring the *PhytoLex* data to the Wikibase, we met with several difficulties which standard Mediawiki plugins could not resolve. Hence, we developed a Django-Wikibase adapter. It serves as an integration layer between Django and Wikibase and allows the implementation of flexible solutions for synchronisation between an existing database and its open-data copy in Wikibase.

We hope the adapter we developed will improve the data transfer from relational databases to the graph of Wikibase.

Acknowledgment

The investigation was carried out for the project "Plants and People in the Russian Empire of the 18th Century: Knowledge and Practice Distribution among Social Classes". The authors are grateful to the European University at St. Petersburg for the support.

References

[1] J. Kesäniemi, M. Koho, & E. Hyvönen. “Using Wikibase for Managing Cultural Heritage Linked Open Data Based on CIDOC CRM”, *European Conference on Advances in Databases and Information Systems*. Cham: Springer, 2022, pp. 542-549.
[2] Virtuoso official website. Web: https://virtuoso.openlinksw.com/.
[3] Hernández D. et al. “Querying wikidata: Comparing sparql, relational and graph databases”, *International Semantic Web Conference*. Cham: Springer, 2016, pp. 88-103.
[4] Malyshev S. et al. “Getting the most out of Wikidata: semantic technology usage in Wikipedia’s knowledge graph” *International Semantic Web Conference.* Cham: Springer, 2018, pp. 376-394.
[5] D. Diefenbach, M.D. Wilde, S. Alipio. “Wikibase as an infrastructure for knowledge graphs: The EU knowledge graph”, *International Semantic Web Conference.* Cham: Springer, 2021, pp. 631-647.
[6] C. Schöch, M. Hinzmann, J. Röttgermann, K. Dietz & A. Klee. “Smart Modelling for Literary History”, *International Journal of Humanities and Arts Computing*, 16.1 (2022), pp. 78-93.
[7] C. Shimizu et al. (2022). “Ontology Design Facilitating Wikibase Integration — and a Worked Example for Historical Data”, *arXiv:2205.14032v1 [cs.AI]* 27 May 2022. Web: https://arxiv.org/pdf/2205.14032.pdf.
[8] Wikidata official website. Wikidata:Development plan. Web: https://www.wikidata.org/wiki/Wikidata:Development_plan.
[9] L. Pintscher, L. Voget, M. Koeppen, E. Aleynikova. *Strategy for the Wikibase Ecosystem*. Web: https://upload.wikimedia.org/wikipedia/commons/c/cc/Strategy_for_Wikibase_Ecosystem.pdf.
[10] Semantic Web official website. Linking Women Editors of Periodicals to the Wikidata Knowledge Graph. Web: https://www.semantic-web-journal.net/content/linking-women-editors-periodicals-wikidata-knowledge-graph-0.
[11] V. Vitari. “Wikidata/Wikibase: A Green Platform Solution for Coworking”, *World Library and Information Congress (WLIC) Materials*. International Federation of Library Associations and Institutions (IFLA), 2022. Web: https://repository.ifla.org/handle/123456789/2049
[12] Wikidata Integrator. Web: https://github.com/SuLab/WikidataIntegrator.
[13] Wikibase. MediaWiki Wikibase extension. Web: https://doc.wikimedia.org/Wikibase/master/php/md_docs_topics_storage.html.

[14] W3C official website. Resource Description Framework (RDF) Model and Syntax Specification. Web: https://www.w3.org/TR/PR-rdf-syntax/.

[15] V. Kolosova, K. Zaytseva, K. Kovalenko. "PhytoLex – the Database of Russian Phytonyms: from Idea to Implementation", *SlaviCorp 2018. 24–26 September 2018. Charles University, Prague. Book of Abstracts,* pp. 88-90. Web: https://slavicorp.ff.cuni.cz/wp-content/uploads/sites/144/2018/09/SlaviCorp2018_Book_of_Abstracts.pdf.

[16] Django official website. Web: https://www.djangoproject.com.

[17] Y. Roskov et al., eds. (2019). *Species 2000 & ITIS Catalogue of Life, 2019 Annual Checklist.* Digital resource at www.catalogueoflife.org/annual-checklist/2019. Species 2000: Naturalis, Leiden, 2019.

[18] PhytoLex official website. Web: https://phytolex.iling.spb.ru.

[19] V.B. Kolosova, K.I. Kovalenko, G.A. Molkov. "The PhytoLex plant name database: results and perspectives", *Diacronia 14*, December 12, 2021, pp. 1-6.

[20] *Slovar′ russkogo jazyka XI–XVII vv.* [*Dictionary of the Russian language of the 11th–17th cc.*]. *Vol. 31.* Moscow: Nauka, 2019, p. 4.

[21] European University at St. Petersburg official website. Center for Machine Learning, Data Analysis, and Statistics. Web: https://eusp.org/en/mast/about.

[22] Transkribus official website. Russian print of the 18 c. (V. Okorokov's Printing House). Web: https://readcoop.eu/model/russian-print-of-the-18-c-v-okorokovs-printing-house/.

[23] Wikimedia official website. The Wikidata revolution is here: enabling structured data on Wikipedia. Web: https://diff.wikimedia.org/2013/04/25/the-wikidata-revolution/.

[24] A. Bielefeldt, J. Gonsior, M. Krötzsch. "Practical linked data access via SPARQL: the case of wikidata", LDOW'2018, April 2018, Lyon, France. Web: https://openreview.net/pdf.

[25] W3C official website. SPARQL 1.1 Query Language. Web: https://www.w3.org/TR/sparql11-query/.

[26] V.R. Benjamins, J. Contreras, O. Corcho, A. Gomez-Perez. *Six challenges for the Semantic Web*. 2002. Web: https://www.researchgate.net/publication/49911494_Six_challenges_for_the_Semantic_Web.

[27] A. Malve, P. Chawan. *A Comparative Study of Keyword and Semantic based Search Engine.* 4. 2015. Web: https://www.researchgate.net/publication/316514673_A_Comparative_Study_of_Keyword_and_Semantic_based_Search_Engine.

[28] WikiData official website. Wikidata:Query Builder. Web: https://www.wikidata.org/wiki/Wikidata:Query_Builder.

[29] WikiData official website. Wikidata:SPARQL query service. Web: https://www.wikidata.org/wiki/Wikidata:SPARQL_query_service

[30] MediaWiki official website. Extension:Page Forms. Web: https://www.mediawiki.org/wiki/Extension:Page_Forms.

[31] MediaWiki official website. Extension:Form. Web: https://www.mediawiki.org/wiki/Extension:Form.

[32] Wikibase solutions official website. Developer log #1 :: WSForm. Web: https://wikibase-solutions.com/developer-logs/wsform.

[33] Django official website. Documentation. Databases. Web: https://docs.djangoproject.com/en/4.1/ref/databases/.

[34] Wikibase official website. Wikibase/DataModel. Web: https://www.mediawiki.org/wiki/Wikibase/DataModel.