Improve DBpedia Track Autumn 2022

A community track, where everybody can participate and contribute in improving existing DBpedia components, in particular the extraction framework, the mappings, the ontology, data quality test cases, new extractors, links and other extensions. Best individual contributions will be acknowledged on the DBpedia website by anointing their WebID/Foaf profile.
The Improve DBpedia track will last **from Sep 21 until Oct 1, 2020, 23:59 Hawaii time**.

Prerequisites

- Register yourself at the forum: http://forum.dbpedia.org
- Join the *#improve-dbpedia* Slack channel via https://dbpedia-slack.herokuapp.com
- Register at the mappings wiki http://mappings.dbpedia.org/index.php/Main_Page#Prerequisites
- Create your WebID: https://github.com/dbpedia/webid

Skills (nice to have, not mandatory)

- Programming skills: Scala/Java, bash, SPARQL, SHACL
- Concepts: HTTP, URI, RDF, HTML, Semantic Web

Table of Contents

Track Task Forces (TFs)

The tasks behind the Improve DBpedia track are grouped into four Task Forces (TFs).

1) **Find-Test-Fix task force** - participants identify an improvement task, write a test and implement the fix.
2) **Open Data Quality task force** - participants implement a method/algorithm which fixes a data quality issue for a particular DBpedia dataset.
3) **Knowledge Extractors task force** - participants improve existing or develop a new knowledge extractor for the DBpedia extraction framework.
4) **Concrete & Complex task force** - participants work on a very concrete tasks and develop a solution, e.g. Create a DBpedia Data Overview.

While each of the task forces are individually defined, very often tasks and work within one task force will partially span/overlap with other task forces.

1. Task Force: Find-Test-Fix (FTF)

Main contacts:

- *Milan Dojchinovski (@m1ci)*
- *Marvin Hofer (@marvinh)*

Description: The Find-Test-Fix task force implements an innovative approach of improving DBpedia by enforcing a test-driven methodology which starts with identifying a problem and documenting it, to fixing and testing it. The FTF task force (i.e. its volunteers) will 1) make use of the syntax reports and the reports from the large-scale validation, 2) identify potential issues for fixing, 3) write a test(s) for each issue, 4) work on the improvement to fix the issue, and finally, 5) make a submission and commit the changes. The developers will follow a test-driven methodology which relies on short development cycles where identified *issues are translated into very specific test cases*, followed by code improvement so that the tests pass.
Detailed description of the Find-Test-Fix task force can be found below in this document.

2. Task Force: Open Data Quality

Main Contact:

- *Tommaso Soru (@tsoru)*
- *Edgard Marx (@emarx)*

Description:
DBpedia derives knowledge from semi-structured sources in Wikipedia. While parts of the extracted information is of relatively high quality, there are portions of data which require further

cleansing and improvement. Within this task force, we invite submissions which will address data quality issues in the extracted DBpedia knowledge graph. Choose the data from the latest DBpedia release published on the DBpedia databus which you would like to work on and implement a method for fixing particular issues. Each submission should be accompanied with description and describe the input dataset, the problem (i.e., identified issues), and the method. Data can be downloaded from the DBpedia Databus manually (https://databus.dbpedia.org/dbpedia.collections/latest-core or using docker/docker-compose or programmatically, for example with the DBpedia Databus Collection Downloader (https://github.com/dbpedia/dbpedia-databus-collection-downloader).
The best submission will be acknowledged on the new DBpedia website and integrated in the future DBpedia releases!
Some potential Data Quality topics:

- Fixing wrong types, datatypes via Semantic Tech (e.g., SHACL)
- Outlier detection via Data Mining (e.g., KG embeddings)
- Fact validation via multiple approaches (e.g., other KGs, Wikipedia text)

Should you be interested in participating in the Data-Quality-related Task Force, feel free to drop a message in the #improve-dbpedia Slack channel!

3. Task Force: Knowledge Extractors

Supervisors:

- *Fabian Hoppe (@fabian)*
- *Mykola Medynskyi (@Mykola Medynskyi)*

Description: Extractors represent the core of the DBpedia Information Extraction Framework. So far, many extractors have been developed for extraction of particular information from different Wikimedia projects. Within this task force we invite submissions which will exclusively focus on 1) extension or fixing existing extractors or 2) implementation of new extractors. Below we present two ideas for new extractors, however, we would be happy if you come up with your own novel ideas for a new extractor or an improvement of an existing extractor. Each submission should be accompanied with description and describe the source for extraction, the implementation and provide an excerpt of the extracted information. The best submission will be acknowledged on the new DBpedia website and the implementation integrated in the extraction framework!

Topic 1: Lists Extractor

Description:
Besides its primary content, i.e. articles on any kind of subject or entities, Wikipedia also contains aggregated "stand-alone lists", i.e. articles that contain lists of entities that have

something in common. Frequently, these lists provide additional information about the listed entities, which are relevant in the context of the given list page, as e.g. the global Alexa Internet page ranking in [1]. This additional information could be utilized to extend the DBpedia knowledge graph. Therefore, the general pattern of a list element has to be detected, based on this general pattern the attribute values extracted and the collected tuples transformed into suitable RDF triples.

Since this topic contains a large research part it is unlikely that it can be wrapped up within the two weeks of the hackathon. Instead the hackathon should kick-off this work and showcase research based on the DBpedia Information Extraction Framework.

[1]: https://en.wikipedia.org/wiki/List_of_social_networking_websites

Topic 2: Wikimedia Commons Extractors

Description:
As you know the Wikimedia Commons is an online repository that contains many different images, sounds, and other files. Each file from it contains information that describes itself. A license is one of these data. So, it would be good to develop a License Extractor for Wikimedia Commons files. This extractor must extract data from Wikimedia Commons file licenses**.** Some notes about Wikimedia Commons files you can see here: https://docs.google.com/document/d/1zSsp51H3yg0xHwvC-cRs6CbmesMvktprjO946UK1Wfw/edit?usp=sharing It would be also great to implement data extraction from nested templates in Wikimedia Commons files infoboxes. This page could possibly help you with the implementation of new Extractor: https://github.com/dbpedia/extraction-framework/wiki/How-to-create-a-new-Extractor

Should you be interested in participating in the Knowledge Extractors TF feel free to drop a message in the #improve-dbpedia Slack channel!

4. Task Force: Concrete & Complex Tasks

Main Contacts:

- *Magnus Knuth (@mgns)*
- *Dimitris Kontokostas (@jimkont)*

Description: This task force encompasses concrete and complex tasks around different topics. More information can be found in the description of the topics below. Each submission should be accompanied with a description of the solution! The best submissions will go live on the new

DBpedia website! Should you have any ideas for a concrete task, please drop a message in the #improve-dbpedia channel.

Topic 1: Create a DBpedia Data Overview

Submission: Results in form of a video to be submitted by Oct 1st, see submission reqs.

Explore the DBpedia dataset and create a catchy overview of the data contained in DBpedia.

Everything from technical information (e.g. data size, mapping details, language interlinks, community contribution statistics, internal and external dataset links) to data content (e.g. topic ranking, knowledge domain coverage, article description granularity) can be unveiled.

Choose which data you like to display and feel free to use your preferred visualization (tables, diagrams, charts, interactive data visualizations, etc.). Preferred datasets are:

- Latest-core / Tiny Diamond
- Any or all Marvin Bot releases
- Largest diamond, with 220 million entities, 1.4 Billion triples. Queryable via linked data or MongoDB (see here for read-only access)

A comparison of the latest and previous releases is also possible to illustrate data extensions, new additions and changes. The best submission will go live on the new DBpedia website!

Ideas for More Topics

Other related ideas (Feel free to add more, for now or the Spring Hackathon):

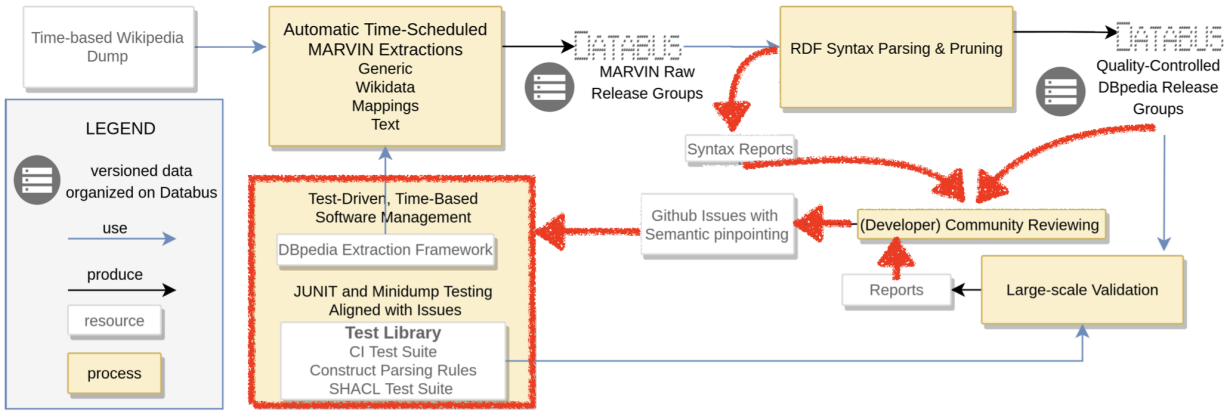
- Single report that reports mappings that are not well defined, domain/range on the wiki, people fix them
- Generating Shape definitions from the ontology and/or learning new ones from DBpedia datasets. SHACL NodeShape definitions allow verification and presentation of data of targeted classes.

Other relevant information

Task Force 1: Find-Test-Fix (detailed description)

The FTF task force covers an important part of the DBpedia release lifecycle. The figure below illustrates the complete release lifecycle. The FTF task force (i.e. its volunteers) will 1) make use of the syntax reports and the reports from the large-scale validation, 2) identify potential issues

for fixing, 3) write a test(s) for each issue, 4) work on the improvement to fix the issue, and finally, 5) make a submission and commit the changes.



The FTF task force will follow a test-driven methodology which relies on short development cycles where identified *issues are translated into very specific test cases*, followed by code improvement so that the tests pass. The methodology will follow four (4) phases:

Phase #1: Pick an issue for improvement

There are two possible options:

i) **Be a "detective"** and identify an issue

There are several sources (not fixed a list) which you can inspect and identify issues for fixing. In the "become detective" section we provide detailed explanations on how to identify an issue.

ii) **Pick an existing issue** from the issue tracker

https://github.com/dbpedia/extraction-framework/issues

We have marked relevant issues with the ci-tests label.

Phase #2: Write a test

Before you start debugging and do code improvement, please write a test first. We have developed a small framework based on *minidumps* tests. Minidumps are small Wikipedia XML dumps which are used to test the extraction framework.

First, *find the URI of the entity* (instance) that relates to the issue and add it to the URIs list.

Then, *write a test*, e.g. a SHACL test to the custom-shacl-tests.ttl file.

After you have written the test, you can continue working on fixing the issue.

Before writing a test, learn:

- How to work with minidumps: http://dev.dbpedia.org/Testing_on_Minidumps

- How to write and execute SHACL tests: http://dev.dbpedia.org/Integrating_SHACL_Tests

Phase #3: Fix the issue

Work on the improvement to fix the issue. The cause of the issue can be related to different sources. For example:

- Bug in the code
  - Investigate the code which relates to the problematic data artifact
  - If the issue is associated with data from the mappings/mappingbased-objects artifact then:
    - visit the databus page for the artifact, e.g. for mappingbased-objects
    - in the Actions section, click on the View Code link which will take you to the code dedicated for the generation of the data artifact.
- Incorrect, or non-existent **mappings** -> improve the mappings
  - http://dev.dbpedia.org/Mappings
  - http://mappings.dbpedia.org/index.php/Main_Page
- Incorrect or non-existent **configuration** -> improve the extraction configuration
  - https://git.informatik.uni-leipzig.de/dbpedia-assoc/marvin-config/-/tree/master/extractionConfiguration
- … to be extended

Phase #4: Commit the changes and test

Introduce the changes as a pull request on the dev branch. At the same time, please test your improvement and make sure your improvement did not lead to a decrease or failure in other parts. Your improvement will be then reviewed by the hacking committee and finally merged with the master branch.

Be a detective and identify a problem

The DBpedia technology stack is huge, however it includes several core technological components:

- DBpedia Extraction Framework: https://github.com/dbpedia/extraction-framework
- DBpedia Mappings:
  - http://mappings.dbpedia.org/index.php/Main_Page
  - Issue tracker: https://github.com/dbpedia/mappings-tracker/issues

- DBpedia Ontology: edited via the mappings, e.g. http://mappings.dbpedia.org/server/ontology/classes/Architect
    - Issue tracker: https://github.com/dbpedia/ontology-tracker/issues
- DBpedia Spotlight service: https://github.com/dbpedia/spotlight-docker/tree/multilingual
- DBpedia Lookup service: https://github.com/dbpedia/lookup-application
- Marving-config - DBpedia monthly release bot: https://git.informatik.uni-leipzig.de/dbpedia-assoc/marvin-config

The best place to start identifying issues is investigating the log files. There are several groups of logs available. More info below.

Extraction log files

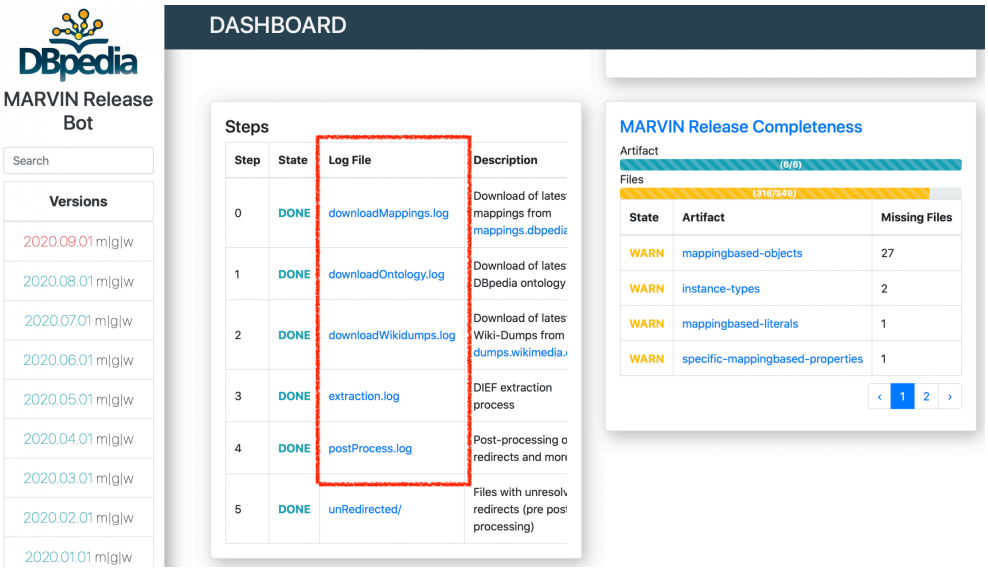
https://release-dashboard.dbpedia.org/?version=2020.08.01

At the https://release-dashboard.dbpedia.org are published logs for the data groups:

- mappings, generic and wikidata;

… and each processing step:

- mappings download,
- ontology download,
- wikidumps download,
- extraction process and
- post-processing.



Parsing reports/logs

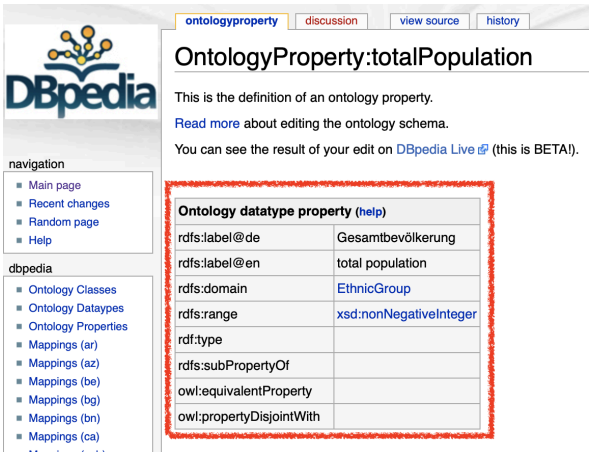
Before a DBpedia release is published, the data is being parsed and checked for syntactical errors. The logs from the parsing are published at http://dbpedia-mappings.tib.eu/parse-reports/

Browse the reports and pick an issue. Currently are published parsing reports for the data artifacts of the mappings and the generic groups. Follows an excerpt of such parsing report: http://dbpedia-mappings.tib.eu/parse-reports/mappings/mappingbased-literals/2020.08.01/mappingbased-literals_lang=en_debug.txt.bz2

<http://dbpedia.org/resource/Bengalis> <http://dbpedia.org/ontology/totalPopulation> "-243000000"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> . # WRN@86 Lexical form '-243000000' not valid for datatype XSD nonNegativeInteger
<http://dbpedia.org/resource/Canis_Major> <http://dbpedia.org/ontology/declination> "-11"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> . # WRN@85 Lexical form '-11' not valid for datatype XSD nonNegativeInteger
<http://dbpedia.org/resource/Circinus> <http://dbpedia.org/ontology/declination> "-55"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> . # WRN@82 Lexical form '-55' not valid for datatype XSD nonNegativeInteger
<http://dbpedia.org/resource/Devonport_City_FC> <http://dbpedia.org/ontology/capacity> "-3168"^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> . # WRN@88 Lexical form '-3168' not valid for datatype XSD nonNegativeInteger
<http://dbpedia.org/resource/Domestic_Manners_of_the_Americans> <http://dbpedia.org/ontology/numberOfPages> "0"^^<http://www.w3.org/2001/XMLSchema#positiveInteger> . # WRN@109 Lexical form '0' not valid for datatype XSD positiveInteger

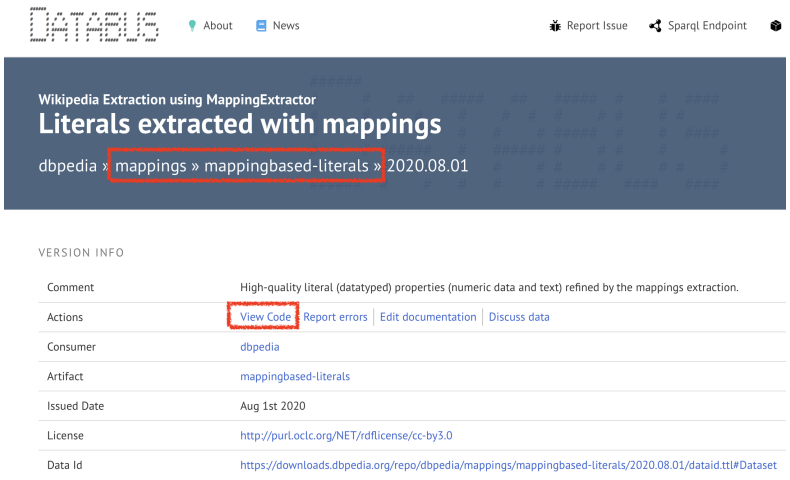
Investigate the problem/cause. In the report, we can see that there is issue with the triple <http://dbpedia.org/resource/Bengalis> <http://dbpedia.org/ontology/totalPopulation> "-243000000" ^^<http://www.w3.org/2001/XMLSchema#nonNegativeInteger> . # WRN@86 Lexical form '-243000000' not valid for datatype XSD nonNegativeInteger

… and the warning message says that the problem is that the value is a negative, however it should be a non-negative (i.e. positive) integer. This constraint is defined in the DBpedia ontology: http://mappings.dbpedia.org/index.php/OntologyProperty:TotalPopulation



Next step would be to check how this information is structured in the infobox of the respective Wikipedia article. The population is indicated as “c. 228-243 million” (see below) and the value is incorrectly parsed (i.e. the value -243 million only considered).

Relate data to code. Next would be to find the code which does the actual parsing. To ease the job of relating data and code, we have referenced data artifacts with the code.
This problematic triple is from the “mappings/mappingbased-literals” data artifact, so we can navigate to the databus page for this artifact https://databus.dbpedia.org/dbpedia/mappings/mappingbased-literals/2020.08.01 and follow the View Code link which points to the class dedicated for the extraction of the particular information. From here, you can further debug and provide a fix.



Before coding and introducing an improvement, **DON’T FORGET TO**:

1. **Open an issue tracker**
    - If you think you need to further discuss with the others:
        - Document and post the issue also at http://forum.dbpedia.org
        - Or ask on the *#improve-dbpedia* slack channel
2. **Write a test using the minidumps**

Useful links

- http://forum.dbpedia.org
- http://dbpedia.slack.com
- http://mappings.dbpedia.org
- https://github.com/dbpedia
- http://databus.dbpedia.org
- http://twitter.com/dbpedia
- https://wiki.dbpedia.org/events
- https://github.com/dbpedia/extraction-framework
- http://dev.dbpedia.org

Communication

For real-time communication lets use the dedicated public **#improve-dbpedia** channel in the Slack DBpedia workspace. If you are not there yet, feel free to join via https://dbpedia-slack.herokuapp.com

- If necessary, we will organize one hour long “Live Q&A” sessions/calls where volunteers can ask any open questions. Don’t be afraid to ask for such a support call!
- Post your questions:
    - in the **#improve-dbpedia** Slack channel, if you think it is necessary a real-world communication - experts from the committee will help answer you questions, or
    - in the **https://forum.dbpedia.org** if your question requires broader discussion.

Are you lost? Write to milan.dojchinovski@informatik.uni-leipzig.de

Hacking Committee

guide, define tasks, answer questions, assist and merge code and contributions

1. Milan Dojchinovski (chair), DBpedia Association, AKSW/KILT
2. Marvin Hofer (co-chair), DBpedia Association, AKSW/KILT
3. Dimitris Kontokostas, Diffbot
4. Mykola Medynskyi, AKSW/KILT

5. Nandana Mihindukulasooriya, MIT-IBM Watson AI Lab
6. Tommaso Soru, Liber AI, DBpedia Association, AKSW
7. Magnus Knuth, eccenca GmbH
8. Diego Moussallem, Paderborn University
9. Johannes Frey, DBpedia Association, AKSW/KILT
10. Denis Streitmatter, AKSW/KILT
11. Fabian Hoppe, German DBpedia/FIZ Karlsruhe
12. Natanael Arndt, AKSW/KILT
13. Edgard Marx, Liber AI, AKSW/Leipzig University of Applied Science (HTWK)
14. Mariano Rico, Spanish DBpedia, Ontology Engineering Group, UPM