Aalto University
School of Science

UNIVERSITY OF HELSINKI

SeCo

HELDIG
Helsinki Centre for Digital Humanities

Creating LOD from databases

Annastiina Ahola, Heikki Rantala

Converting Data to RDF

- **Converting good, clean structured data is pretty easy, but data is rarely that good.**
- **What do you need to consider?**
  - Controlled vocabularies
  - URIs (identifiers)
  - Data models / ontologies
  - …

Tools for Converting Data to RDF

- **Programming libraries**
  - RDFLib (Python), Jena (Java)...
- **RDF Mapping Language (RML)**
- **OpenRefine**
- **Lots of other tools…**

Example data: ArtSampo

- **Original data from Finnish National Gallery's data dump available online: https://www.kansallisgalleria.fi/en/api-sovelluskehittajille**
- **Information on over 80,000 art works and objects and nearly 9,000 persons related to them**
  - e.g., collections from Ateneum Art Museum, Museum of Contemporary Art Kiasma and Sinebrychoff Art Museum
- **Data dump in JSON format**

ArtSampo transformation process

- **Data conversion from JSON to RDF done using the Python library RDFLib (https://rdflib.readthedocs.io/)**
- **The script iterates over each object present in the JSON and extracts information on**
  a. the art work/object itself,
  b. any possible multimedia attached to the object, and
  c. the people listed with each object
- **Transformation is done on a very simple level: Things that have IDs in the data are modeled as objects (e.g., people) but other things are kept as simple literals (e.g., organisations)**

Transformation process

original JSON


```
{
        "objectId": 406873,
        "responsibleOrganisation": "Kansallisgalleria / Ateneumin
taidemuseo",
        "dimensions": [
                { "unit": "cm", "sortLnu": 2, "measureType": { "en":
"image", "fi": "kuva", "id": 210514, "sv": "bild", "type": "measure-type",
"updated": null }, "measurements": [ 30, 24.5 ]},
                { "unit": "cm", "sortLnu": 2, "measureType": { "en":
"paper", "fi": "lehti", "id": 210516, "sv": "papper", "type": "measure-type",
"updated": null }, "measurements": [ 49.5, 35 ] },
                { "unit": "cm", "sortLnu": 2, "measureType": { "en":
"plate", "fi": "laatta", "id": 210515, "sv": "plåt", "type": "measure-type",
"updated": null }, "measurements": [ 30, 24.5 ] }
        ],
        "collection": { "fi": "Rolando ja Siv Pieraccinin kokoelma", "en":
"Rolando and Siv Pieraccini Collection", "sv": "Rolando och Siv Pieraccinis
samling" },
        "title": { "en": "", "fi": "Rappuja nousevia miehiä ja naisia", "sv":
"" },
        ...
```


RDF

```
fng:o406873 a art:Object ;
        art:artist <http://ldf.fi/artsampo/fng/persons/p59954> ;
        art:classification "grafiikka" ;
        art:collection "Rolando ja Siv Pieraccinin kokoelma" ;
        art:dimensions [ art:length 30.0 ; art:measureType "laatta" ; art:unit
                "cm" ;  art:width 24.5 ],
                [ art:length 30.0 ; art:measureType "kuva" ; art:unit "cm" ;
                art:width  24.5 ],
                [ art:length 49.5 ; art:measureType "lehti" ; art:unit "cm" ;
                art:width 35.0 ] ;
        art:inventoryNumber "A-2011-82" ;
        art:keyword "man"@en, "men"@en, "stairs"@en, "woman"@en, "women"@en,
        "miehet"@fi, "mies"@fi, "nainen"@fi, "naiset"@fi, "portaat"@fi ;
        art:objectId 406873 ;
        art:responsibleOrganisation "Kansallisgalleria / Ateneumin taidemuseo" ;
        art:title "Rappuja nousevia miehiä ja naisia"@fi ;
        art:yearFrom 1973 ;
        skos:prefLabel "Rappuja nousevia miehiä ja naisia" .
```

CSV example: OperaSampo

1724	5001051	Dorle	von Wendt	von Wendt Dorle						
1725	14322463	Theodor	Görcke	Görcke Theodor						
1726	3563392	Axel Mauritz	Hansson	Hansson Axel Mauritz	7.7.1869	9.7.1911	Horten, Norge	Hornbaek, Danmark	<?xml version='1.0' encoding='UTF-8'?><root available-locales="fi_FI,en_US," default-locale="fi_FI"><AdditionalInfo language-id="en_US">Swedish actor\nSource: Svenskt porträttgalleri, XXI (Stockholm 1897), s. 42.\nWikipedia</AdditionalInfo></root>	Svensk skådespelare\nSource: Svenskt porträttgalleri, XXI (Stockholm 1897), s. 42.\nWikipedia
1727	3649477	Aline	Kumlander	Kumlander Aline						

2781	6357628	6357627	role.opera-singer
2782	3581517	3563392	role.opera-singer
2783	3626756	3624491	role.director

foc_PersonRole.csv

foc_Person.csv

```
personId|firstName|lastName|displayName|dateOfBirth|dateOfDeath|placeOfBirth|placeOfDeath|additionalInfo|editorNotes
...
3563392|Axel Mauritz|Hansson|Hansson Axel Mauritz|7.7.1869|9.7.1911|Horten, Norge|Hornbaek, Danmark|"<?xml version='1.0' encoding='UTF-8'?><root available-locales=""fi_FI,en_US,"" default-locale=""fi_FI""><AdditionalInfo language-id=""en_US"">Swedish actor\nSource: Svenskt porträttgalleri, XXI (Stockholm 1897), s. 42.\nWikipedia</AdditionalInfo></root>"|Svensk skådespelare\nSource: Svenskt porträttgalleri, XXI (Stockholm 1897), s. 42.\nWikipedia
...
```

foc_Person.csv

```
personRoleId|personId|role
...
3581517|3563392|role.opera-singer
...
```

foc_PersonRole.csv

```
ops:persons_3563392 a scop:Person ;
        scop:additionalInfo "Swedish actor<br>Source: Svenskt porträttgalleri,
                XXI (Stockholm 1897), s. 42.<br>Wikipedia"@en ;
        scop:dateOfBirth "7.7.1869" ;
        scop:dateOfDeath "9.7.1911" ;
        scop:editorNotes "Svensk skådespelare<br>Source: Svenskt porträttgalleri,
                XXI (Stockholm 1897), s. 42.<br>Wikipedia" ;
        scop:placeOfBirth "Horten, Norge" ;
        scop:placeOfDeath "Hornbaek, Danmark" ;
        scop:role ops:occupation_roles_opera-singer ;
        skos:prefLabel "Hansson Axel Mauritz"@fi ;
        foaf:firstName "Axel Mauritz" ;
        foaf:surname "Hansson" .
```

RDF

Example code: ArtSampo transformation

```
# import necessary libraries
import json # options for CSV data: pandas, csv
from rdflib import Namespace, URIRef, Literal, Graph, RDF, RDFS, XSD, FOAF


# define namespaces for easier use
# (e.g., ART.Object = <http://ldf.fi/schema/artsampo/Object>)
SKOS = Namespace('http://www.w3.org/2004/02/skos/core#')
FNG = Namespace('http://ldf.fi/artsampo/fng/')
ART = Namespace('http://ldf.fi/schema/artsampo/')
...
```

Example code: ArtSampo transformation

```
# create a new graph
g = Graph()


# bind relevant namespaces
graph.bind('rdf', 'http://www.w3.org/1999/02/22-rdf-syntax-ns#')
graph.bind('skos', 'http://www.w3.org/2004/02/skos/core#')
graph.bind('fng', 'http://ldf.fi/artsampo/fng/')
graph.bind('art', 'http://ldf.fi/schema/artsampo/')
...
```

Example code: ArtSampo transformation

```
...
# open your source data file and load its data
with open('objects.json') as fng_file:
    data = json.load(fng_file) # for CSV: pandas.read_csv, csv.reader

# iterate over the objects (or lines for CSV)
for item in data:
    # extract data from the object …
    ...
```

Example code: ArtSampo transformation

```
for item in data:
    # formulate a URI using the object's 'objectId' field
    art_object = URIRef(str(FNG)+'o'+str(item['objectId']))
    # add the triple '<URI> rdf:type art:Object' to the graph
    g.add((art_object, RDF.type, ART.Object))
    # extract other information through, e.g., custom
    # functions for each different field of data ...
    add_labels(g, item, art_object)
    ...
```

Example code: ArtSampo transformation

```
...
# serialize the graph when you're done to save it (in this
# case a file named 'fng.ttl' to the folder ttl/)
g.serialize('ttl/fng.ttl', format='turtle')
```

Local Fuseki setup for querying data

- **You can use a Apache Jena Fuseki (https://jena.apache.org/documentation/fuseki2/) SPARQL server with Docker for serving your data locally**
  - This enables you to query your own created data locally with SPARQL similarly to an online endpoint and use it for analysis
    - *More on how to utilize SPARQL with LOD later in "Using the LOD service via SPARQL and Notebooks, data analysis, network analysis, and visualizations"*

Improving RDF

- **Mapping vocabularies**
  - My “Helsinki” concept is the same as your “Helsinki” concept
  - My “Viking sword” concept is a broad match to your “Petersen E type sword” concept
- **Mapping my model to some general ontology**
  - Rules to map my data model to, for example, CIDOC CRM
- **Perfection can be the enemy of good! RDF and data in general does not have to be perfect to be useful!**

Conclusions

- **Your data drives the process: What *is* in your data?**
  - Use actual example cases from your data to help you think about the data model and conversion process – your data doesn't have to be "beautiful" to be useful and you can always adjust things (e.g., different modeling decisions) later
- **Work in steps: Start small and iterate**
  - You can first include only some properties in your conversion
    - *Does the output look correct? Are there any issues with the data that might need some preprocessing, for example?*
  - Continue expanding your conversion process to include other properties when the previous ones are OK

Conclusions

- **Validating your data**
  - Setting up an user interface could help you see mistakes in data (e.g., missing values or something being overrepresented)
  - Formal validation with SHACL, ShEx
  - Domain-specific validation
    - *8-star model – is your data truthful?*
      - e.g., person participating in an event after their death or a start date being later in time than an end date