Knowledge Engineering Mini-Project

Using two data sources, a remote API and a local turtle file converted from a CSV, in one federated query

COLLABORATORS

- Fayssal EL ANSARI - M2 Informatique
- EL ANSARI Fayssal - M2 Informatique

Contents

Tasks

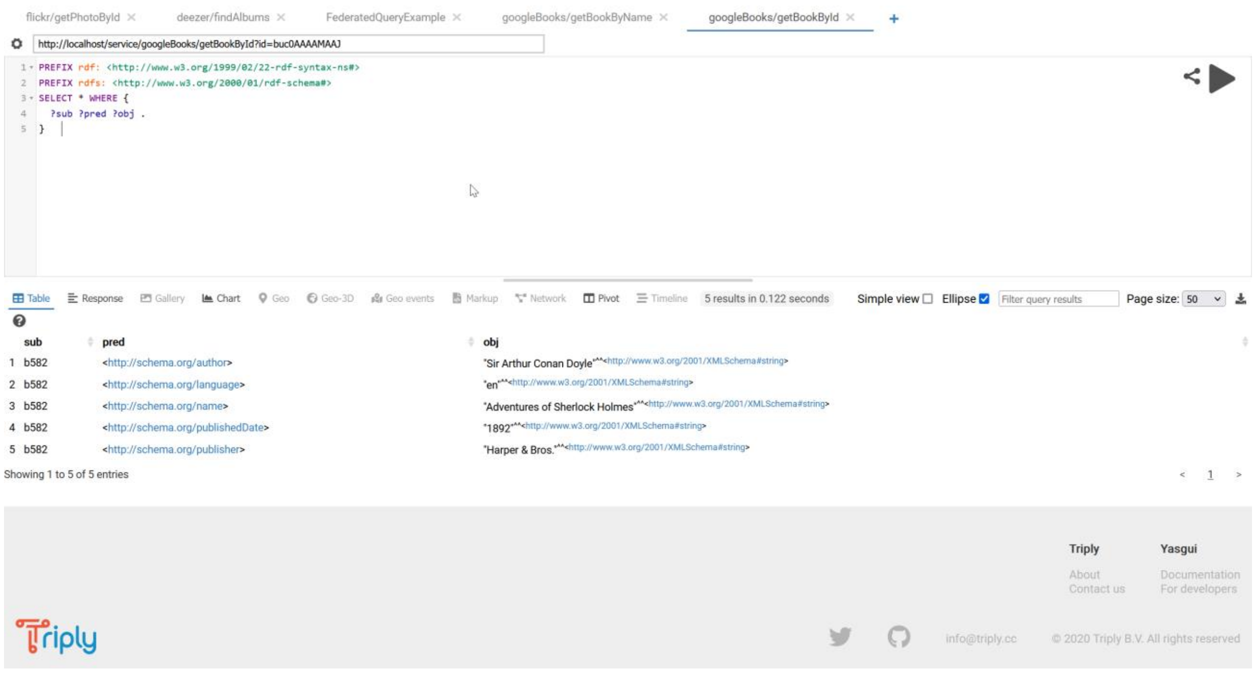
- ☑ Choose `SPARQL` API microservice.
    - `Google's books API`
- ☑ Create boilerplate code for `GoogleBooks micro-service`
- ☑ create a first version `microservice`
    - ☑ getBookById
        - ☑ modify config.ini
        - ☑ modify construct.sparql
        - ☑ modify profile.jsonld
    - ☑ getBookByName
        - ☑ modify config.ini
        - ☑ modify construct.sparql
        - ☑ modify profile.jsonld
- ☑ Align data to `DBpedia` (or schema.org or wikidata..)
- ☑ Choose `CSV` file.
    - file downloaded from `https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks`
- ☑ Lift the `CSV` file using a CSVW mapping and a chosen tool.
- ☑ Write a `SPARQL` federated query involving both data sources.
- ☐ (optional) create a second version `microservice`
    - ☐ add more entities
    - ☑ add an actual id to every node representing a book in the graph
    - ☑ make the id clickable to download data in the form of `ttl` (just like in provided examples)
    - ☐ demo html page

To be finished before the **08/01/2024**.

PROJECT DESCRIPTION

We'll start off by choosing a free and comprehensive API that could be dereferenced using SPARQL, after some research we have decided to use `Google's books API`, this page describes how to use this api in detail https://developers.google.com/books/docs/v1/getting_started , an example of a JSON response is provided in the annexe

We'll be using two endpoints the first one to get one singular book by id, and the second one to get all books with a name similar to the name passed as a parameter.



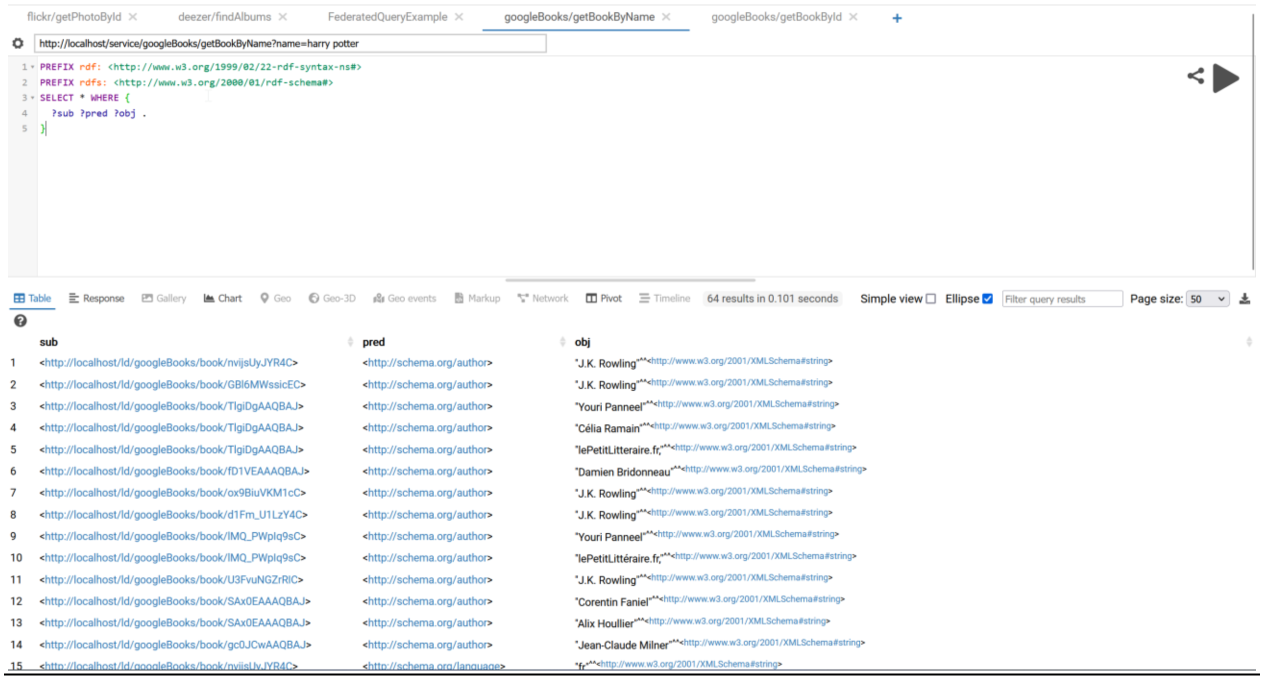
http://localhost/service/googleBooks/getBookById?id=buc0AAAAMAAJ

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT * WHERE {
  ?sub ?pred ?obj .
}
```

5 results in 0.122 seconds

	sub	pred	obj
1	b582	<http://schema.org/author>	"Sir Arthur Conan Doyle"^^<http://www.w3.org/2001/XMLSchema#string>
2	b582	<http://schema.org/language>	"en"^^<http://www.w3.org/2001/XMLSchema#string>
3	b582	<http://schema.org/name>	"Adventures of Sherlock Holmes"^^<http://www.w3.org/2001/XMLSchema#string>
4	b582	<http://schema.org/publishedDate>	"1892"^^<http://www.w3.org/2001/XMLSchema#string>
5	b582	<http://schema.org/publisher>	"Harper & Bros."^^<http://www.w3.org/2001/XMLSchema#string>

Showing 1 to 5 of 5 entries



http://localhost/service/googleBooks/getBookByName?name=harry potter

```
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
SELECT * WHERE {
  ?sub ?pred ?obj .
}
```

64 results in 0.101 seconds

	sub	pred	obj
1	<http://localhost/ld/googleBooks/book/nvijsUyJYR4C>	<http://schema.org/author>	"J.K. Rowling"^^<http://www.w3.org/2001/XMLSchema#string>
2	<http://localhost/ld/googleBooks/book/GBl6MWssicEC>	<http://schema.org/author>	"J.K. Rowling"^^<http://www.w3.org/2001/XMLSchema#string>
3	<http://localhost/ld/googleBooks/book/TlgiDgAAQBAJ>	<http://schema.org/author>	"Youri Panneel"^^<http://www.w3.org/2001/XMLSchema#string>
4	<http://localhost/ld/googleBooks/book/TlgiDgAAQBAJ>	<http://schema.org/author>	"Célia Ramain"^^<http://www.w3.org/2001/XMLSchema#string>
5	<http://localhost/ld/googleBooks/book/TlgiDgAAQBAJ>	<http://schema.org/author>	"lePetitLitteraire.fr,"^^<http://www.w3.org/2001/XMLSchema#string>
6	<http://localhost/ld/googleBooks/book/fD1VEAAAQBAJ>	<http://schema.org/author>	"Damien Bridonneau"^^<http://www.w3.org/2001/XMLSchema#string>
7	<http://localhost/ld/googleBooks/book/ox9BiuVKM1cC>	<http://schema.org/author>	"J.K. Rowling"^^<http://www.w3.org/2001/XMLSchema#string>
8	<http://localhost/ld/googleBooks/book/d1Fm_U1LzY4C>	<http://schema.org/author>	"J.K. Rowling"^^<http://www.w3.org/2001/XMLSchema#string>
9	<http://localhost/ld/googleBooks/book/lMQ_PWplq9sC>	<http://schema.org/author>	"Youri Panneel"^^<http://www.w3.org/2001/XMLSchema#string>
10	<http://localhost/ld/googleBooks/book/lMQ_PWplq9sC>	<http://schema.org/author>	"lePetitLittéraire.fr,"^^<http://www.w3.org/2001/XMLSchema#string>
11	<http://localhost/ld/googleBooks/book/U3FvuNGZrRIC>	<http://schema.org/author>	"J.K. Rowling"^^<http://www.w3.org/2001/XMLSchema#string>
12	<http://localhost/ld/googleBooks/book/SAx0EAAAQBAJ>	<http://schema.org/author>	"Corentin Faniel"^^<http://www.w3.org/2001/XMLSchema#string>
13	<http://localhost/ld/googleBooks/book/SAx0EAAAQBAJ>	<http://schema.org/author>	"Alix Houllier"^^<http://www.w3.org/2001/XMLSchema#string>
14	<http://localhost/ld/googleBooks/book/gc0JCwAAQBAJ>	<http://schema.org/author>	"Jean-Claude Milner"^^<http://www.w3.org/2001/XMLSchema#string>
15	<http://localhost/ld/googleBooks/book/nvijsUyJYR4C>	<http://schema.org/language>	"fr"^^<http://www.w3.org/2001/XMLSchema#string>

Ontology

To align this data to Schema.org we'll take a couple of entities such as:

- https://schema.org/identifier
- http://schema.org/name
- http://schema.org/author
- https://schema.org/aggregateRating
- https://schema.org/isbn
- https://schema.org/isbn13
- https://schema.org/Language
- https://schema.org/numberOfPages
- https://schema.org/ratingCount
- https://schema.org/reviewCount
- https://schema.org/datePublished
- http://schema.org/publisher

CSV

The purpose of the complementary CSV file can be either to add information that was not provided by Google's API, or to provide second source of the same information.
The CSV file was downloaded from `https://www.kaggle.com/datasets/jealousleopard/goodreadsbooks`. It contains 11k different entries already pre-processed.

Before starting the CSV2RDF conversion process, we'll need to start out with a working example first, there are many examples in the official swirrl/csv2rdf git repository under the path:
https://github.com/Swirrl/csv2rdf/tree/master/w3c-csvw/tests/

we start off by working on test011 which contains 4 separate files, the metadata.json, tree.csv, result.json and result.ttl. supposedly we should be able to create the result.ttl with a command similar to that provided in the course. Using csv2rdf on minimal mode with books.csv only and apache any23 default csv extractor gives similar results, but using csv2rdf with a metadata.json that references the books.csv file produces an empty books.ttl file.

After a thorough investigation, we finally realised that the reason why the produced .ttl file was empty is because the tool is unable to read lines which contain " or ' characters. Could be because of the way the csv file is being read, and it was empty because the tool writes into the output file at the end of the conversion and not one by one (to avoid having multiple open/close tasks probably). If the writing process was being done one by one or if there was a verbose/debugging mode it would have been easier to identify the root cause of the problem.

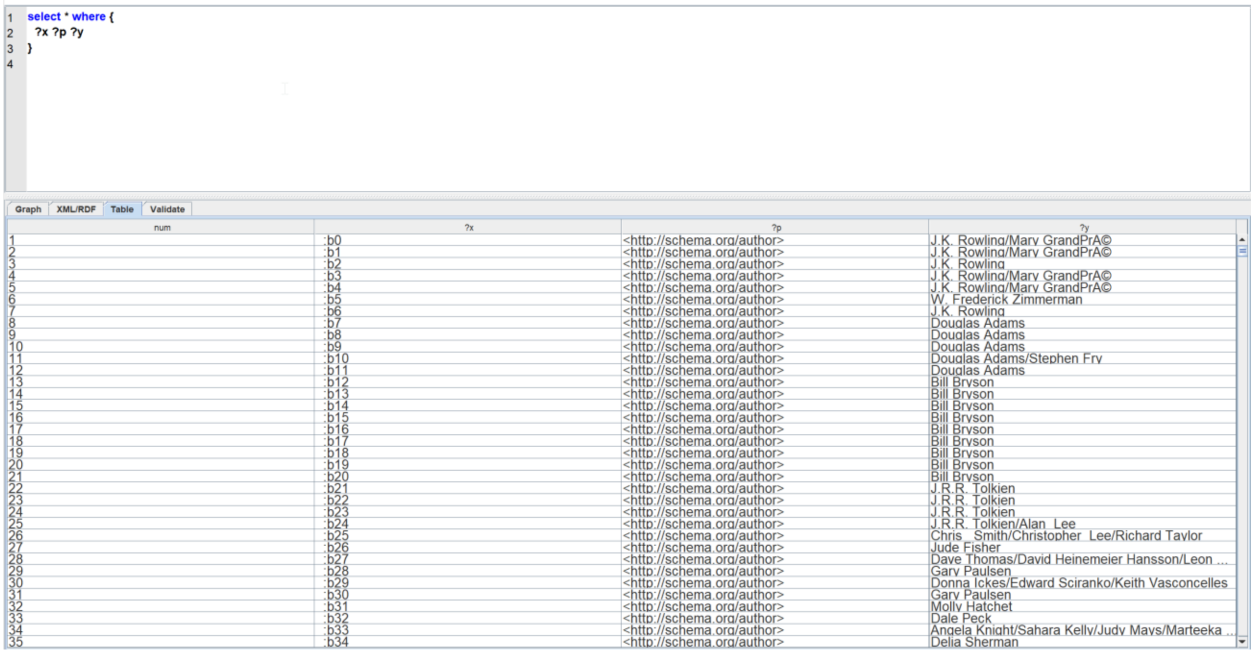
CSV Details

CSV field	Description	API field	notes
bookID	A unique Identification number for each book. An incremental number assigned by the creator of the CSV file	-	Is different from the API's id field.
title	The name under which the book was published	volumeInfo.title	this is the linking point between the API and the csv file
authors	Names of the authors of the book. Multiple authors are delimited with -	volumeInfo.authors	
average_rating	The average rating of the book received in total	volumeInfo.averageRating	
isbn	Another unique number to identify the book, the International Standard Book Number	volumeInfo.industryIdentifiers.identifier	In the industryIdentifiers list when the type is "ISBN_10" the corresponding identifier is the equivalent field.
isbn13	A 13-digit ISBN to identify the book, instead of the standard 11-digit ISBN	volumeInfo.industryIdentifiers.identifier	In the industryIdentifiers list when the type is "ISBN_13" the corresponding identifier is the equivalent field.
language_code	Helps understand what is the primary language of the book. For instance, eng is standard for English	volumeInfo.language	

num_pages	Number of pages the book contains	volumeInfo.pageCount	The values are not always the same
ratings_count	Total number of ratings the book received	volumeInfo.ratingsCount	Mos of the times this field is not present
text_reviews_count	Total number of written text reviews the book received	-	

Lifting CSV

We first need to define a mapping, we'll be using CSVW (csv on the web) language. In our metadata file we have added the previous third-party ontology references, and `aboutUrl` which uses the title of the book. Since the csv provided in the csv is only an incremental number different from the id provided in the API we've decided to use the name of the book (title) as a primary key. We also added the type of each field when possible.



```
select * where {
  ?x ?p ?y
}
```

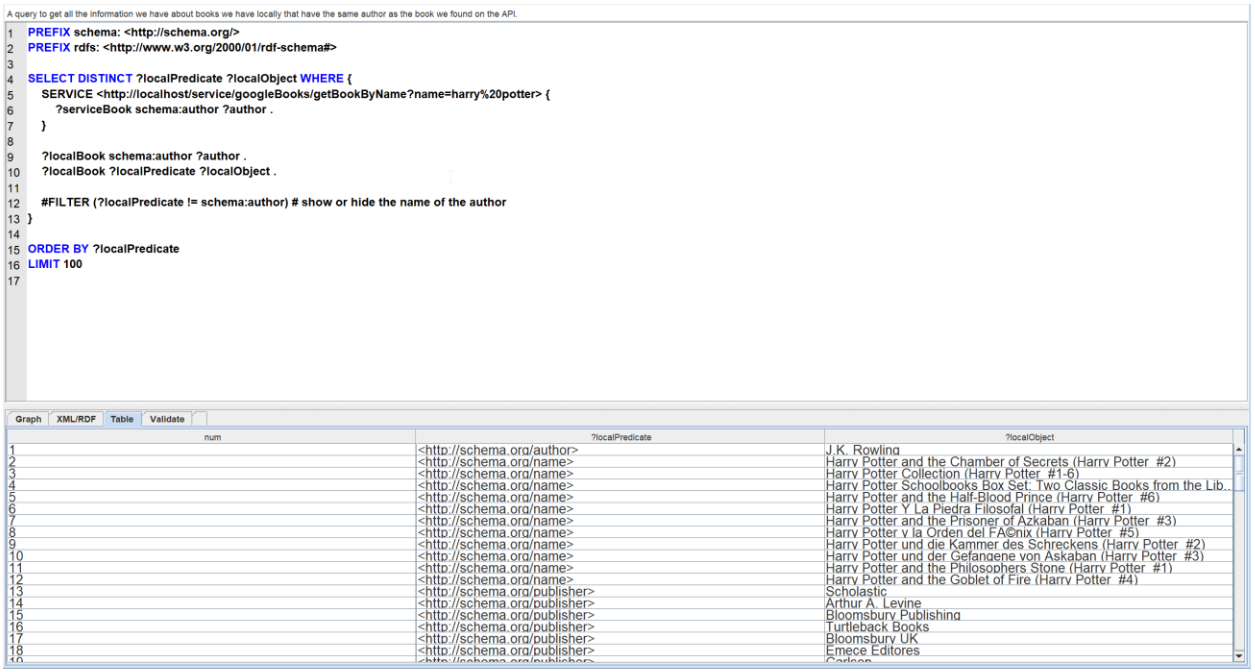
num	?x	?p	?y
1	:b0	<http://schema.org/author>	J.K. Rowling/Mary GrandPrÃ©
2	:b1	<http://schema.org/author>	J.K. Rowling/Mary GrandPrÃ©
3	:b2	<http://schema.org/author>	J.K. Rowling
4	:b3	<http://schema.org/author>	J.K. Rowling/Mary GrandPrÃ©
5	:b4	<http://schema.org/author>	J.K. Rowling/Mary GrandPrÃ©
6	:b5	<http://schema.org/author>	W. Frederick Zimmerman
7	:b6	<http://schema.org/author>	J.K. Rowling
8	:b7	<http://schema.org/author>	Douglas Adams
9	:b8	<http://schema.org/author>	Douglas Adams
10	:b9	<http://schema.org/author>	Douglas Adams
11	:b10	<http://schema.org/author>	Douglas Adams/Stephen Fry
12	:b11	<http://schema.org/author>	Douglas Adams
13	:b12	<http://schema.org/author>	Bill Bryson
14	:b13	<http://schema.org/author>	Bill Bryson
15	:b14	<http://schema.org/author>	Bill Bryson
16	:b15	<http://schema.org/author>	Bill Bryson
17	:b16	<http://schema.org/author>	Bill Bryson
18	:b17	<http://schema.org/author>	Bill Bryson
19	:b18	<http://schema.org/author>	Bill Bryson
20	:b19	<http://schema.org/author>	Bill Bryson
21	:b20	<http://schema.org/author>	Bill Bryson
22	:b21	<http://schema.org/author>	J.R.R. Tolkien
23	:b22	<http://schema.org/author>	J.R.R. Tolkien
24	:b23	<http://schema.org/author>	J.R.R. Tolkien
25	:b24	<http://schema.org/author>	J.R.R. Tolkien/Alan Lee
26	:b25	<http://schema.org/author>	Chris Smith/Christopher Lee/Richard Taylor
27	:b26	<http://schema.org/author>	Jude Fisher
28	:b27	<http://schema.org/author>	Dave Thomas/David Heinemeier Hansson/Leon ...
29	:b28	<http://schema.org/author>	Gary Paulsen
30	:b29	<http://schema.org/author>	Donna Ickes/Edward Sciranko/Keith Vasconcelles
31	:b30	<http://schema.org/author>	Gary Paulsen
32	:b31	<http://schema.org/author>	Molly Hatchet
33	:b32	<http://schema.org/author>	Dale Peck
34	:b33	<http://schema.org/author>	Angela Knight/Sahara Kelly/Judy Mays/Marteeka ...
35	:b34	<http://schema.org/author>	Delia Sherman

Result of a query on the lifted CSV

Federated Query

The federated query we've decided to use has as a general purpose to compare what we have in our local library with the result of a search on the web (API call).

The given query gets the books that have similar names to "harry potter" then looks for all books that have the same authors on the local converted csv file.



A query to get all the information we have about books we have locally that have the same author as the book we found on the API.

```
PREFIX schema: <http://schema.org/>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

SELECT DISTINCT ?localPredicate ?localObject WHERE {
   SERVICE <http://localhost/service/googleBooks/getBookByName?name=harry%20potter> {
      ?serviceBook schema:author ?author .
   }

   ?localBook schema:author ?author .
   ?localBook ?localPredicate ?localObject .

   #FILTER (?localPredicate != schema:author) # show or hide the name of the author
}

ORDER BY ?localPredicate
LIMIT 100

```

num	?localPredicate	?localObject
1	<http://schema.org/author>	J.K. Rowling
2	<http://schema.org/name>	Harry Potter and the Chamber of Secrets (Harry Potter #2)
3	<http://schema.org/name>	Harry Potter Collection (Harry Potter #1-6)
4	<http://schema.org/name>	Harry Potter Schoolbooks Box Set: Two Classic Books from the Lib..
5	<http://schema.org/name>	Harry Potter and the Half-Blood Prince (Harry Potter #6)
6	<http://schema.org/name>	Harry Potter Y La Piedra Filosofal (Harry Potter #1)
7	<http://schema.org/name>	Harry Potter and the Prisoner of Azkaban (Harry Potter #3)
8	<http://schema.org/name>	Harry Potter y la Orden del FA©nix (Harry Potter #5)
9	<http://schema.org/name>	Harry Potter und die Kammer des Schreckens (Harry Potter #2)
10	<http://schema.org/name>	Harry Potter und der Gefangene von Askaban (Harry Potter #3)
11	<http://schema.org/name>	Harry Potter and the Philosophers Stone (Harry Potter #1)
12	<http://schema.org/name>	Harry Potter and the Goblet of Fire (Harry Potter #4)
13	<http://schema.org/publisher>	Scholastic
14	<http://schema.org/publisher>	Arthur A. Levine
15	<http://schema.org/publisher>	Bloomsbury Publishing
16	<http://schema.org/publisher>	Turtleback Books
17	<http://schema.org/publisher>	Bloomsbury UK
18	<http://schema.org/publisher>	Emece Editores

References

- https://www.w3.org/TR/csvw-ucr/
- CSV2RDF: https://github.com/Swirrl/csv2rdf/
- https://www.w3.org/TR/2015/REC-csv2rdf-20151217/

Annexe

JSON response

a sample of its response is as follows, the first element of the items list:

```
{
    "kind": "books#volume",
    "id": "buc0AAAAMAAJ",
    "etag": "Jb2BwbsTZIA",
    "selfLink": "https://www.googleapis.com/books/v1/volumes/buc0AAAAMAAJ",
    "volumeInfo": {
        "title": "Adventures of Sherlock Holmes",
        "authors": [
            "Sir Arthur Conan Doyle"
        ],
        "publisher": "Harper & Bros.",
        "publishedDate": "1892",
        "description": "Presenting 12 tales starring the legendary British detective Sherlock Holmes, this
1892 book is Arthur Conan Doyle's first short-story collection. The mystery compilation includes some of
Holmes's finest cases with his dutiful sidekick, Doctor Watson, most notably \"A Scandal in Bohemia,\" in
which Holmes matches wits with the crafty former lover of a European king. Also featured is \"The
Adventure of the Red-Headed League,\" a study in misdirection that unfolds to become a much larger scheme.
The stories, initially published in the Strand Magazine, are essential reading for Holmes fans.",
        "readingModes": {
            "text": true,
            "image": true
        },
        "pageCount": 307,
        "printedPageCount": 360,
        "dimensions": {
            "height": "20.00 cm"
        },
        "printType": "BOOK",
        "averageRating": 4.5,
        "ratingsCount": 485,
        "maturityRating": "NOT_MATURE",
        "allowAnonLogging": false,
        "contentVersion": "7.5.15.0.full.3",
        "panelizationSummary": {
            "containsEpubBubbles": false,
            "containsImageBubbles": false
        },
        "imageLinks": {
            "smallThumbnail":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=5&edge=curl&imgtk=AF
LRE71AsKhoBnnPtp7CDC6MFaf10pRUkJyfaxq_Zau6wZGcidUMSH1vMr3InAUatC8m8JDiP1OUpYWxA41up8N0VCWJZKiMJ9RgBMZz1ov-
G1phK09NXu6tLGp81aYjEr9cNvhNpkvY&source=gbs_api",
            "thumbnail":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=1&edge=curl&imgtk=AF
LRE70P15brdAFvkW5BNXFVWCvhyUIQMB5Ft0c0iZ0Mz4uW8Em5A6y0PqTmDVzzUkguOp_dyIUGIrHcflAvqRWeOFHxv8quoxN_8ZsbcoER
ifE-nrNjjZoQHJSS9aZD6dHBd9EybILO&source=gbs_api",
            "small":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=2&edge=curl&imgtk=AF
LRE715BROhKm0jhKBhgS5WC6-DdujLA5ATUT0pOf1BPxE6K4ECPQfGtnymPeWQI9yNeqi3pE4_CMAtZZVaR5M2R5N-
KaMseAWjWhRQR72PSiv22Yb9NvD7VaAtddIivzfI4rPG98rD&source=gbs_api",
            "medium":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=3&edge=curl&imgtk=AF
LRE72ON1WnAm5Fz0YCRJSkXcZDVNvBCsXb7WBLeWwR4ZBAEgsN6FqBOHCRb_gJhNwvdekYTE3ZDwYShrWFci9n4HePArZmeS1hb31aQiOF
XsxCJkvC9Zur0Q2vEr3IHy02ezoMf94a&source=gbs_api",
            "large":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=4&edge=curl&imgtk=AF
LRE72b5v-rjKpqJ9dLu203QHLO_mFNS6hOTlD1BQRmawyAqE9sVwS99DUyT7A7FM6sjzy_8qgC3tJkjHz-
yNibxkLjxi3ElWJDvvEkf_cUUpn3rIMOEuwDtEYCu4b4Kov0RkJm1953&source=gbs_api",
            "extraLarge":
"http://books.google.com/books/content?id=buc0AAAAMAAJ&printsec=frontcover&img=1&zoom=6&edge=curl&imgtk=AF
LRE70CVyVAQvWDFwLMKSkFWa4DnhmJoECAi_p47IbRY4jRsQFo2c7WGydQmeAoovhDkQjzPqMmDQb214BPpHMlgGBYrGIxjTuRzJbmsEs7
J9ruc92TE8jg2wrxEkoPLwrLSfADg79q&source=gbs_api"
        },
        "language": "en",
        "previewLink": "http://books.google.fr/books?id=buc0AAAAMAAJ&hl=&source=gbs_api",
        "infoLink": "https://play.google.com/store/books/details?id=buc0AAAAMAAJ&source=gbs_api",
        "canonicalVolumeLink": "https://play.google.com/store/books/details?id=buc0AAAAMAAJ"
    },
    "layerInfo": {
        "layers": [
            {
                "layerId": "geo",
                "volumeAnnotationsVersion": "28"
            }
        ]
    },
    "saleInfo": {
        "country": "FR",
        "saleability": "FREE",
        "isEbook": true,
        "buyLink": "https://play.google.com/store/books/details?id=buc0AAAAMAAJ&rdid=book-
buc0AAAAMAAJ&rdot=1&source=gbs_api"
    },
    "accessInfo": {
        "country": "FR",
        "viewability": "ALL_PAGES",
        "embeddable": true,
        "publicDomain": true,
        "textToSpeechPermission": "ALLOWED",
        "epub": {
            "isAvailable": true,
            "downloadLink":
"http://books.google.fr/books/download/Adventures_of_Sherlock_Holmes.epub?id=buc0AAAAMAAJ&hl=&output=epub&
source=gbs_api"
        },
        "pdf": {
            "isAvailable": true,
            "downloadLink":
"http://books.google.fr/books/download/Adventures_of_Sherlock_Holmes.pdf?id=buc0AAAAMAAJ&hl=&output=pdf&si
g=ACfU3U3UFzY0zsHRr3M-FLDDrmM0rrWf4Q&source=gbs_api"
        },
        "webReaderLink": "http://play.google.com/books/reader?id=buc0AAAAMAAJ&hl=&source=gbs_api",
        "accessViewStatus": "FULL_PUBLIC_DOMAIN",
        "quoteSharingAllowed": false
    }
}
```

Provided Questions

1. Figure out a use case to integrate data from 2 non-RDF sources:

- Web API that you will query by writing a SPARQL micro-service:

The API must NOT be those given in example during the TD, Deezer and MusicBrainz, unless you chose a different service and target vocabulary.
Align the data generated on a vocabulary that is relevant for your use case, such as Schema.org, DBpedia, Wikidata or any other one you can find on http://lov.linkeddata.es.
If you do not find any existing relevant vocabulary, create one of your own (RDFS or OWL).

- A CSV file

Lift the file to RDF using a CSVW mapping and the tool of your choice, for example csv2rdf
Align the data generated on a vocabulary that is relevant for your use case, ideally the same as for the SPARQL micro-service
If you do not find a public CSV file that matches your use case, you are allowed to write one. In that case, the grade iwll take into consideration how difficult it is to write the final federated query.

2. Write a SPARQL federated query that involves both data sources: load the RDF file in Corese, and query it together with the SPARQL micro-service.
SPARQL federated queries are written with the SERVICE clause, check the spefication:
https://www.w3.org/TR/sparql11-federated-query/

3. Upload the result of your work as a Zip file containing:

A PDF file describing succinctly your use case, the API and CSV file you have selected, and the vocabulary(ies) that you have selected or created (which classes and properties you used).
a folder sparql-micro-services with the code of the SPARQL micro-service (config.ini, profile.jsonld, construct.sparql)
a folder csvw with: the csv file, the CSVW mapping, and the result file in Turtle generated with csv2rdf.
a screenshot of Yasgui or Corese GUI where you query individually each source with a simple "construct where {?s ?p ?p}".
a screenshot of Yasgui or Corese GUI where you execute the SPARQL federated query (that can be either a SELECT or CONSTRUCT query).

The grade will take into account:

- The quality of the mapping to the target vocabulary
- The challenge of reconciling the two sources: if the CSV file and the API represent exactly the same information and you just gather the results, that's not challenging.