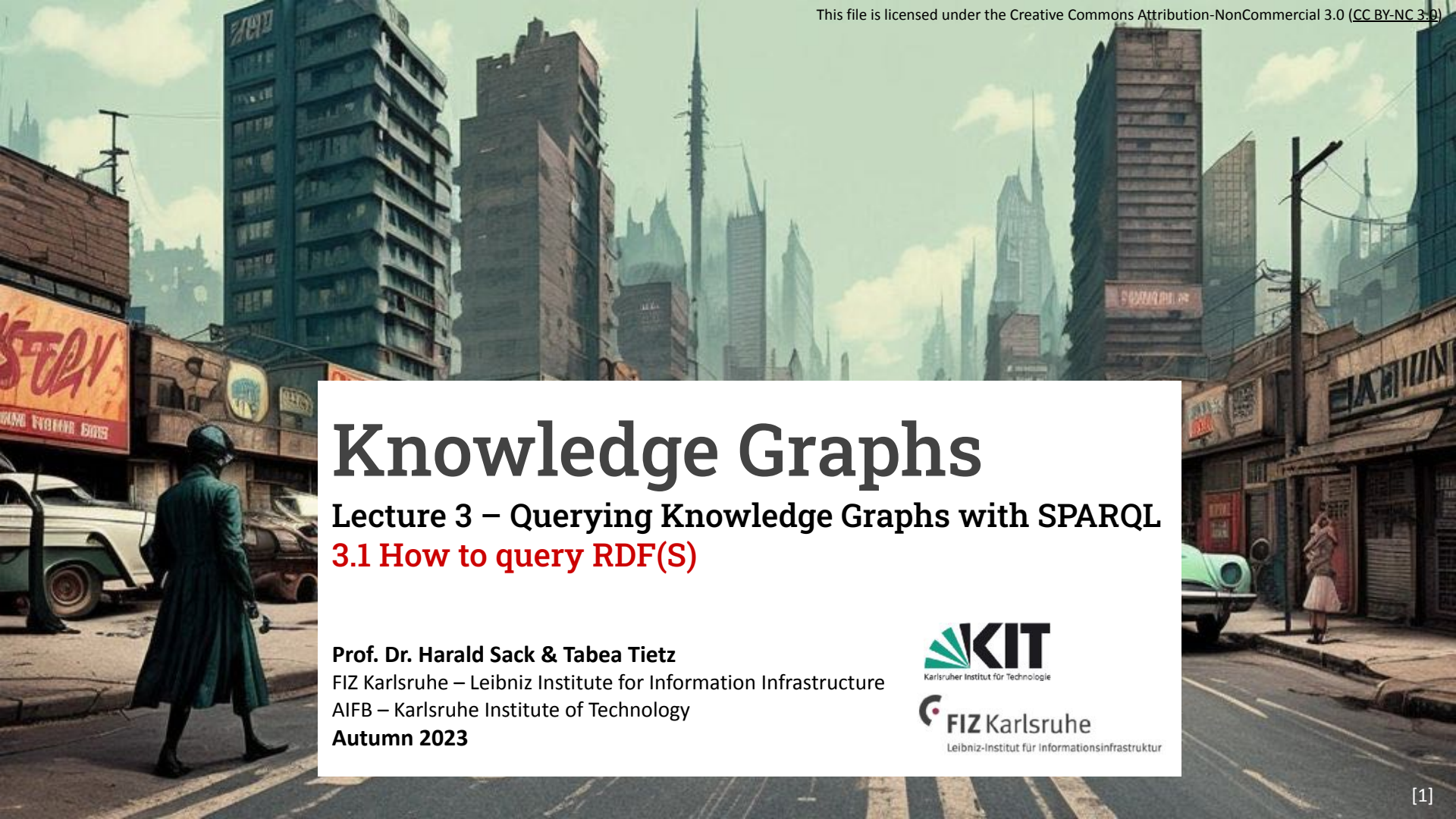


Knowledge Graphs

Lecture 3 – Querying Knowledge Graphs with SPARQL

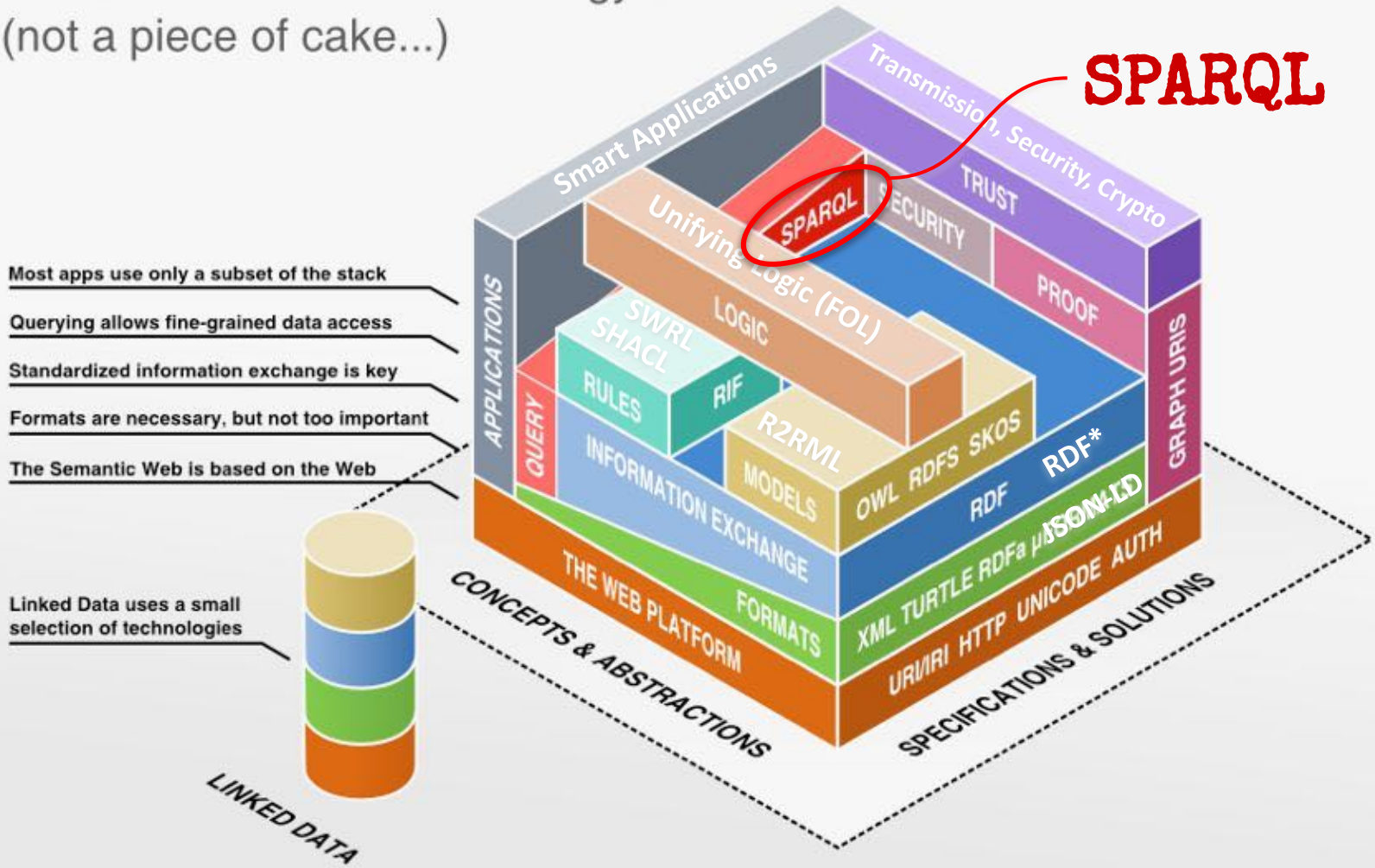
3.1 How to query RDF(S)

Prof. Dr. Harald Sack & Tabea Tietz
FIZ Karlsruhe – Leibniz Institute for Information Infrastructure
AIFB – Karlsruhe Institute of Technology
Autumn 2023

Knowledge Graphs

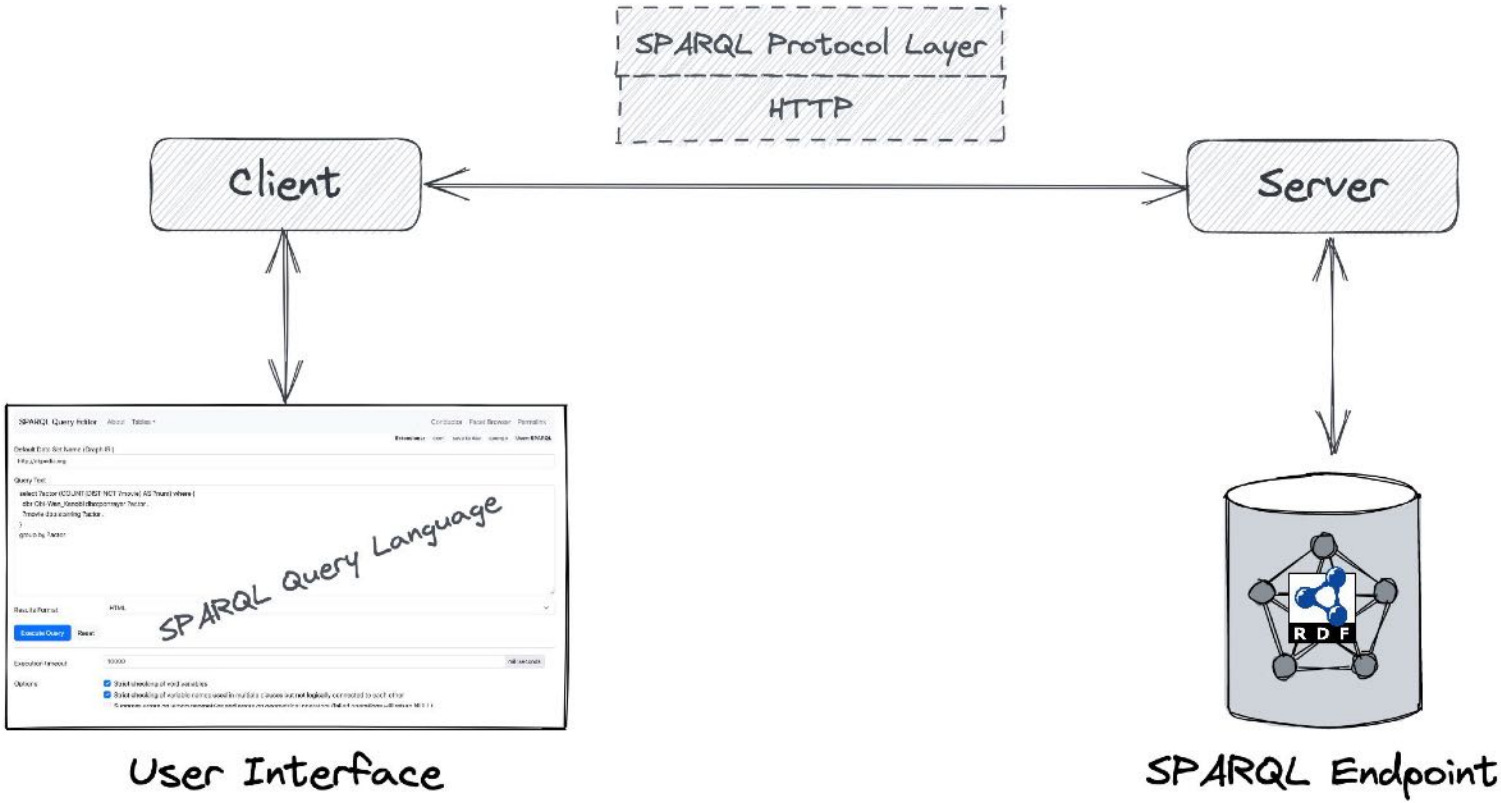
Lecture 3: Querying Knowledge Graphs with SPARQL

The Semantic Web Technology Stack
(not a piece of cake...)

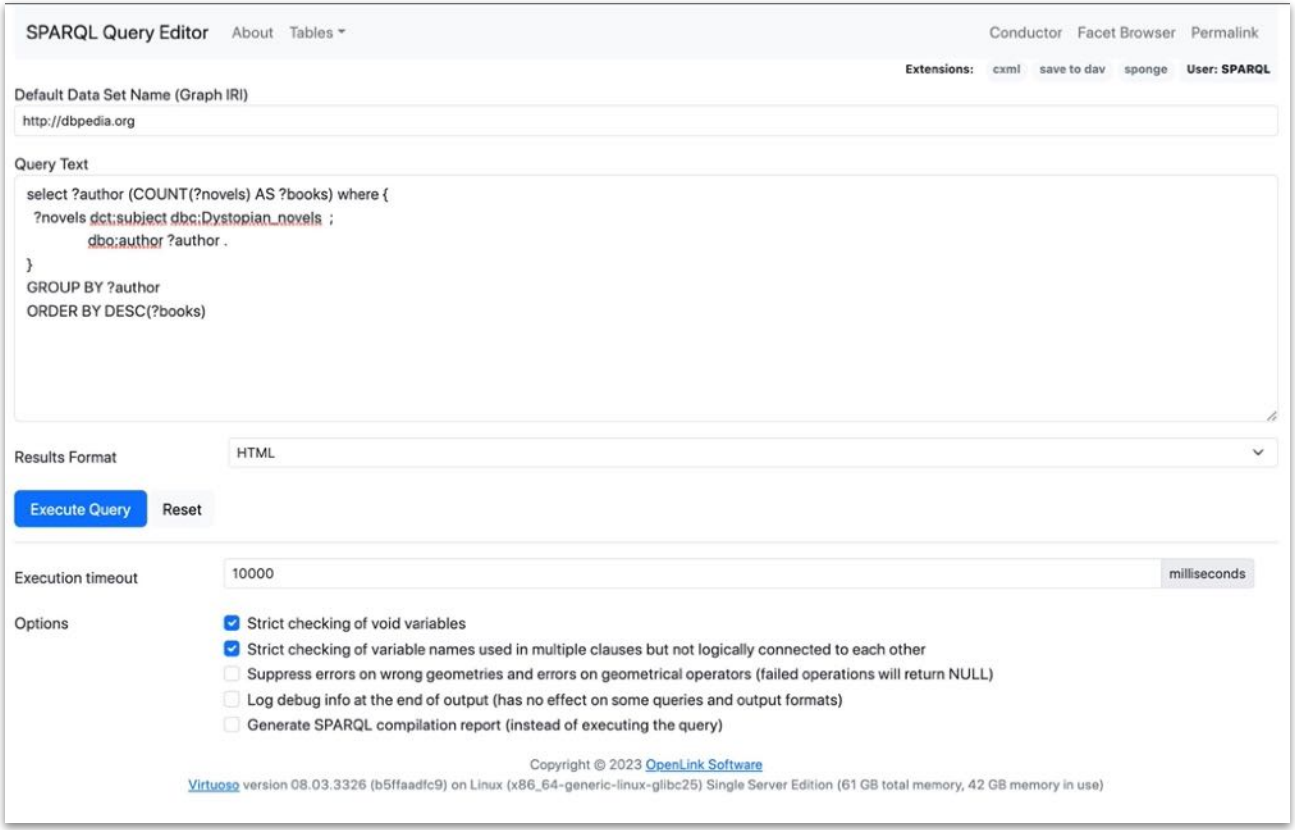


[2]

SPARQL – A Query Language for Knowledge Graphs

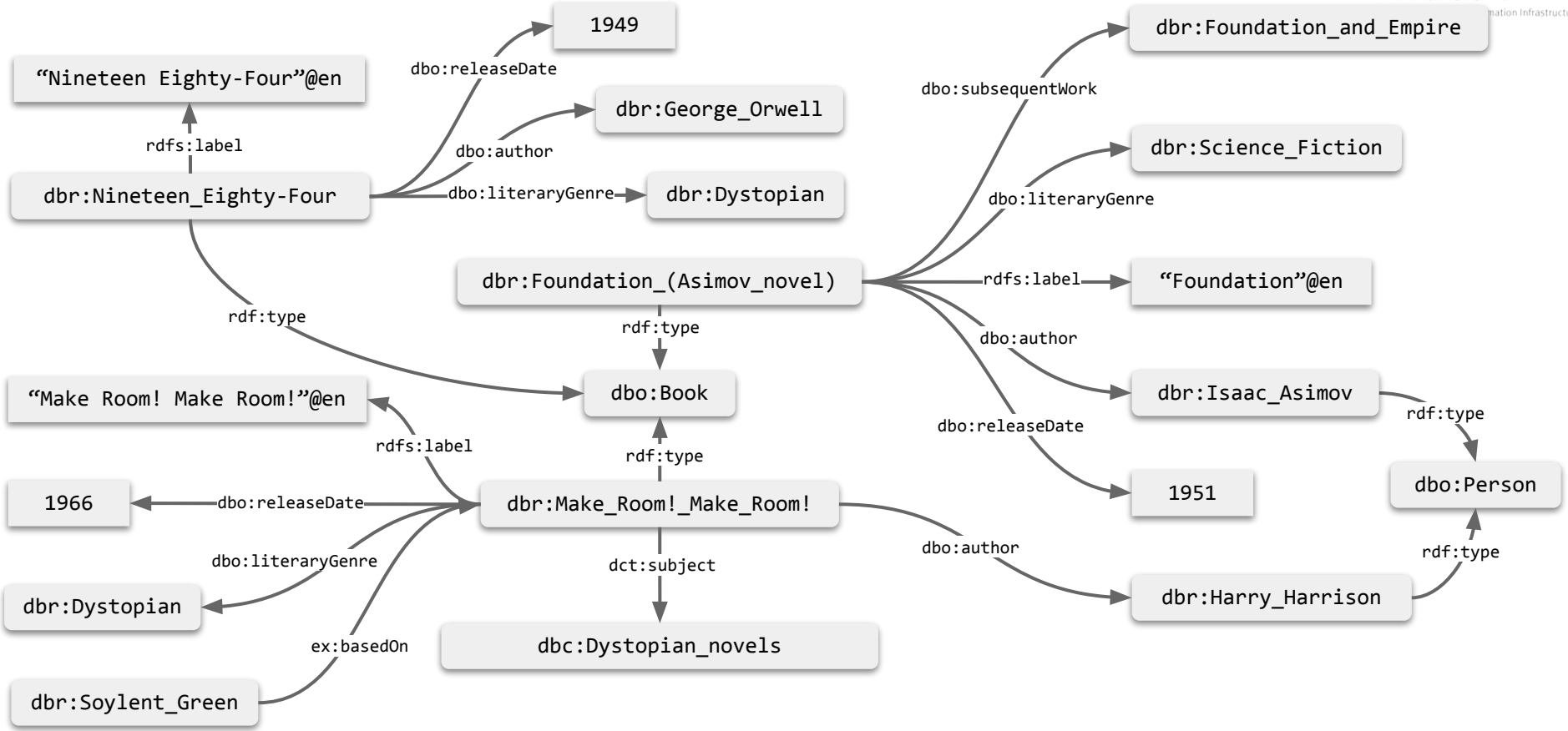


SPARQL – Endpoint Example

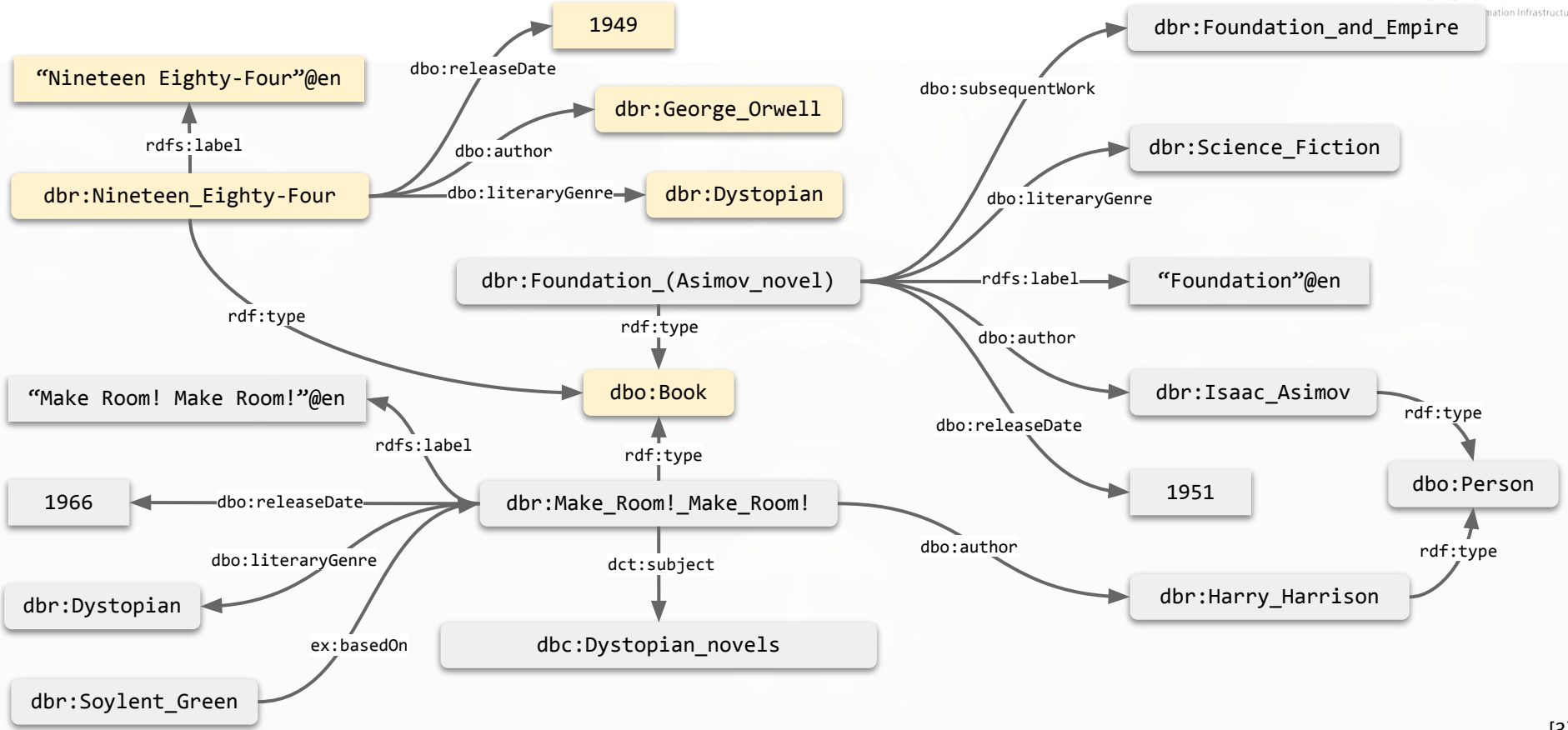


http://dbpedia.org/sparql

Querying an RDF-based Knowledge Graph

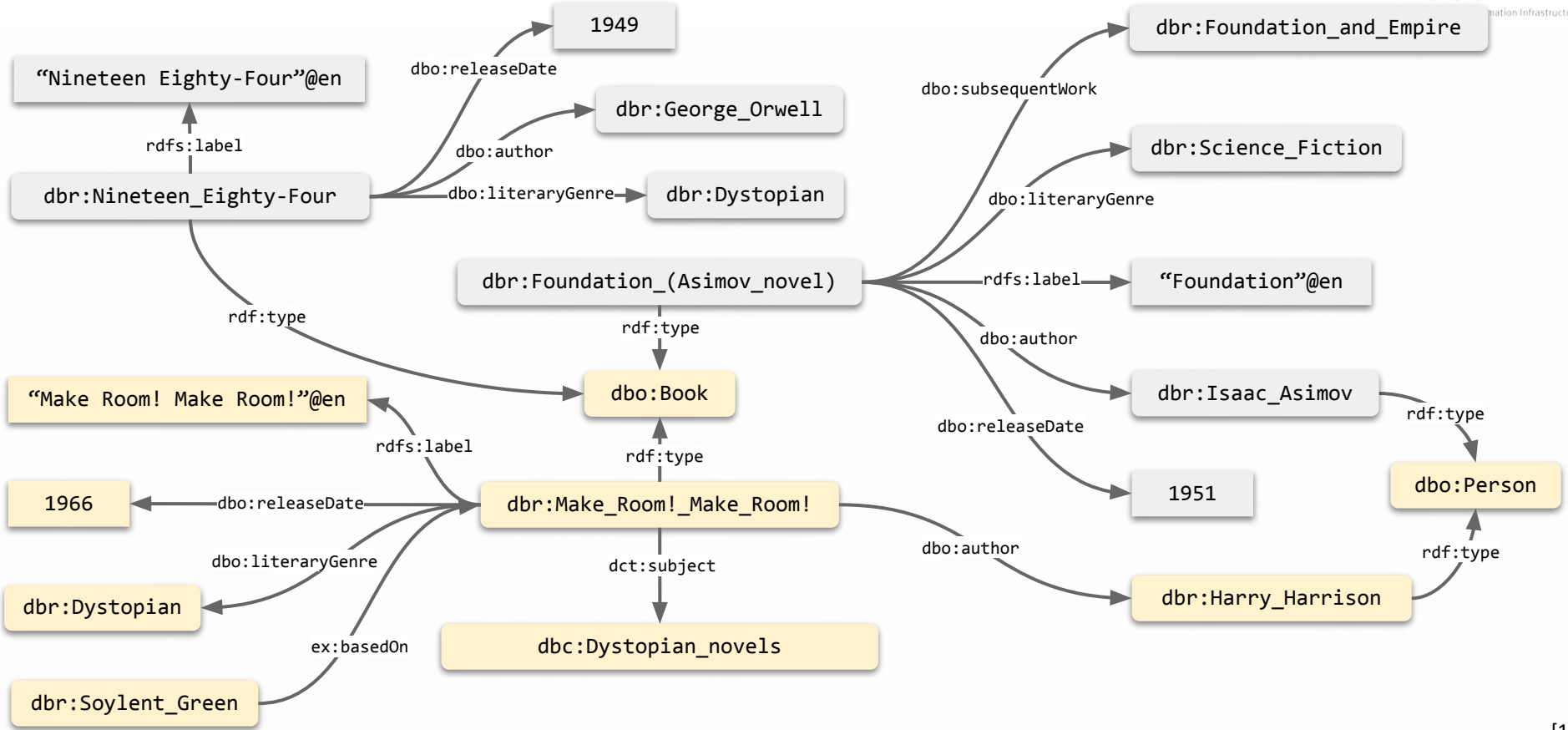


Querying an RDF-based Knowledge Graph



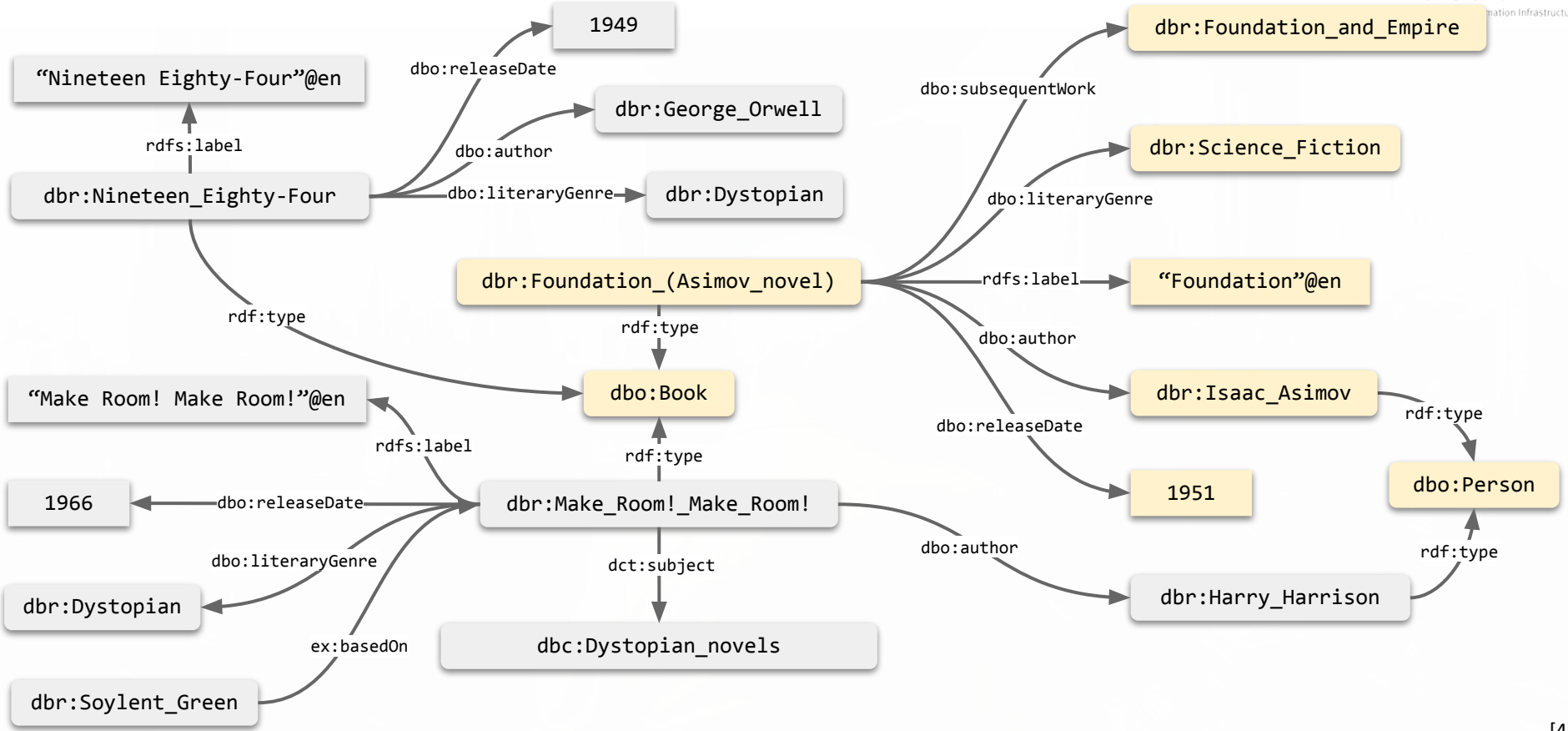
[3]

Querying an RDF-based Knowledge Graph



[1]

Querying an RDF-based Knowledge Graph



[4]

For Queries We Need Variables

- SPARQL **Variables** are bound to RDF terms,

  e.g. **?title, ?author, ?date**

- In the same way as in SQL,

  a **Query for variables** is performed via **SELECT statement**,

  e.g. **SELECT ?title ?author ?date** SPARQL Query

- A SELECT statement returns query results as a **table**.

?title	?author	?date
Nineteen Eighty-Four	George Orwell	1948
Foundation (Novel)	Isaac Asimov	2006
Make Room! Make Room!	Harry Harrison	1966

SPARQL Result

SPARQL Graph Pattern Matching

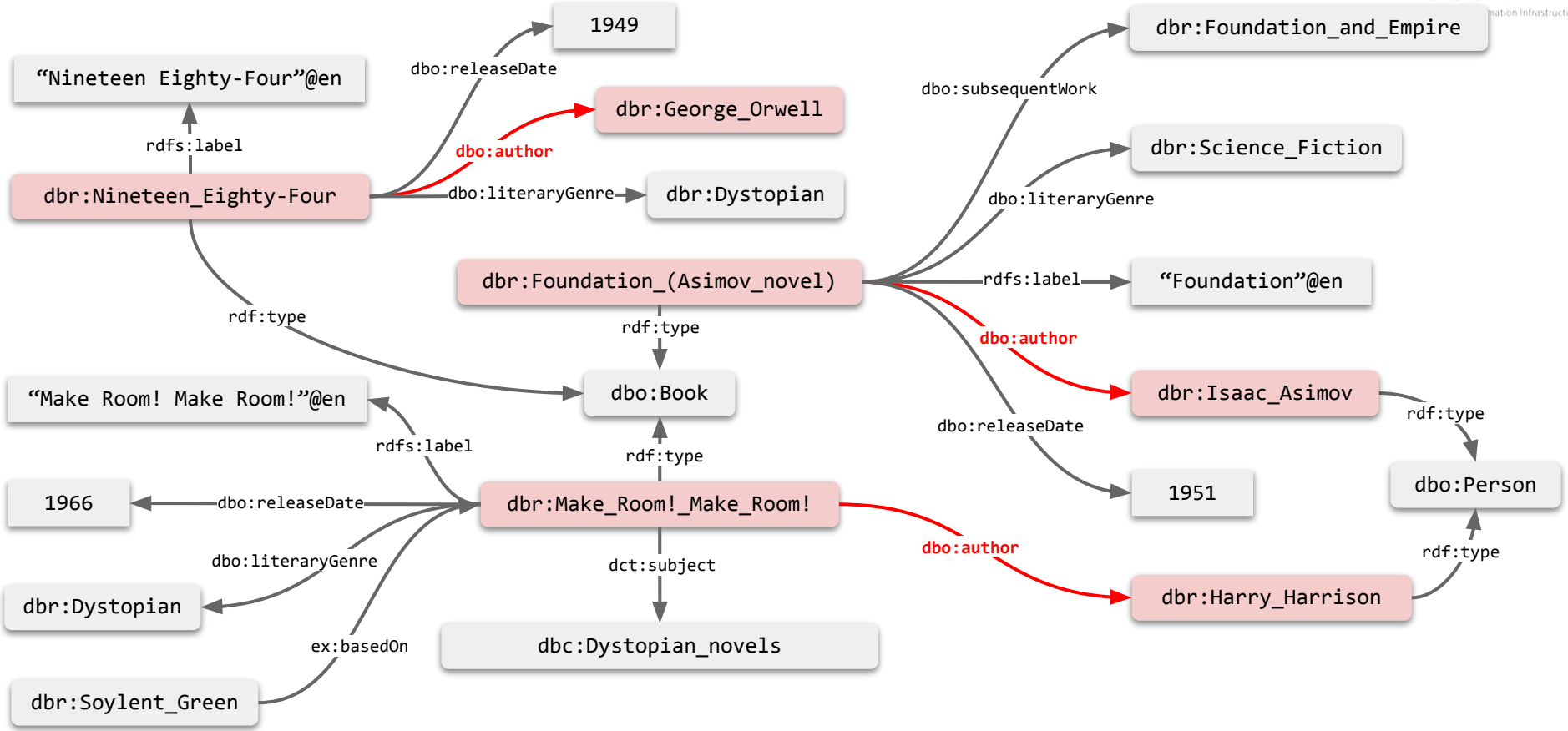
- SPARQL is based on
  (1) **RDF Turtle serialization** and (2) **basic graph pattern matching**.
- A **Graph Pattern** (**Triple Pattern**) is a RDF Triple that contains variables at any arbitrary place (Subject, Property, Object).

  Graph Pattern = Turtle + Variables
- Example:
  Look for books and their authors (via property dbo:author):

```
?book dbo:author ?author .
```

variables

Querying an RDF-based Knowledge Graph



SPARQL Complex Graph Pattern Matching

- SPARQL Graph Pattern can be combined to form **complex (conjunctive) queries** for RDF graph traversal.

- *Find books, their authors, and their literary genres:*

```
?book dbo:author ?author .
?book dbo:literaryGenre ?genre .
```

the same book(s)

SPARQL Complex Graph Pattern Matching

- SPARQL Graph Pattern can be combined to form **complex (conjunctive) queries** for RDF graph traversal.

- *Given a specific book URI, find its author(s), the birthplace(s) of its author(s), including the number of population of the birthplace(s):*

```
dbr:Brave_New_World dbo:author          ?author .
?author             dbo:birthPlace      ?birthplace .
?birthplace         dbo:populationTotal ?population .
```

the same author(s)

the same birthplace(s)

SPARQL Query

```
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo:  <http://dbpedia.org/ontology/>

SELECT ?author_name ?title

FROM <http://dbpedia.org/>

WHERE {
     ?author rdf:type dbo:Writer .
     ?author rdfs:label ?author_name .
     ?author dbo:notableWork ?work .
     ?work rdfs:label ?title .
}
```

specifies namespaces

specifies output variables

specifies graph to be queried

specifies graph pattern to be matched

Example:
search for all **authors** and the **titles** of their **notable works**:

query SPARQL endpoint

SPARQL Query

```
PREFIX :     <http://dbpedia.org/resource/>
PREFIX rdf:  <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo:  <http://dbpedia.org/ontology/>

SELECT ?author_name ?title

FROM <http://dbpedia.org/>

WHERE {
      ?author rdf:type dbo:Writer .
      ?author rdfs:label ?author_name .
      ?author dbo:notableWork ?work .
      ?work rdfs:label ?title .
}
ORDER BY ASC (?author_name)
LIMIT 100
OFFSET 10
```

solution sequence modifiers

Example:
search for all **authors** and the **titles** of their **notable works**: **ordered by authors** in **ascending order** and **limit** the results to **the first 100 results** starting the list at **offset 10** position:

query SPARQL endpoint

SPARQL Filter Constraints

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>

SELECT ?author_name ?title ?pages
FROM <http://dbpedia.org/>
WHERE {
       ?author rdf:type dbo:Writer .
       ?author rdfs:label ?author_name .
       ?author dbo:notableWork ?work .
       ?work dbo:numberOfPages ?pages
       FILTER (?pages > 500) .
       ?work rdfs:label ?title .
} LIMIT 100
```

specifies constraints for the result

Example:
search for all **authors** and the **titles** of their **notable works**: that have **more than 500 pages** and **limit** the results to **the first 100**

FILTER expressions contain operators and functions

query SPARQL endpoint

SPARQL Unary Operators

Operator	Type(A)	Result Type
`!A`	xsd:boolean	xsd:boolean
`+A`	numeric	numeric
`-A`	numeric	numeric
`BOUND(A)`	variable	xsd:boolean
`isURI(A)`	RDF term	xsd:boolean
`isBLANK(A)`	RDF term	xsd:boolean
`isLITERAL(A)`	RDF Term	xsd:boolean
`STR(A)`	literal/URL	simple literal
`LANG(A)`	literal	simple literal
`DATATYPE(A)`	literal	URI

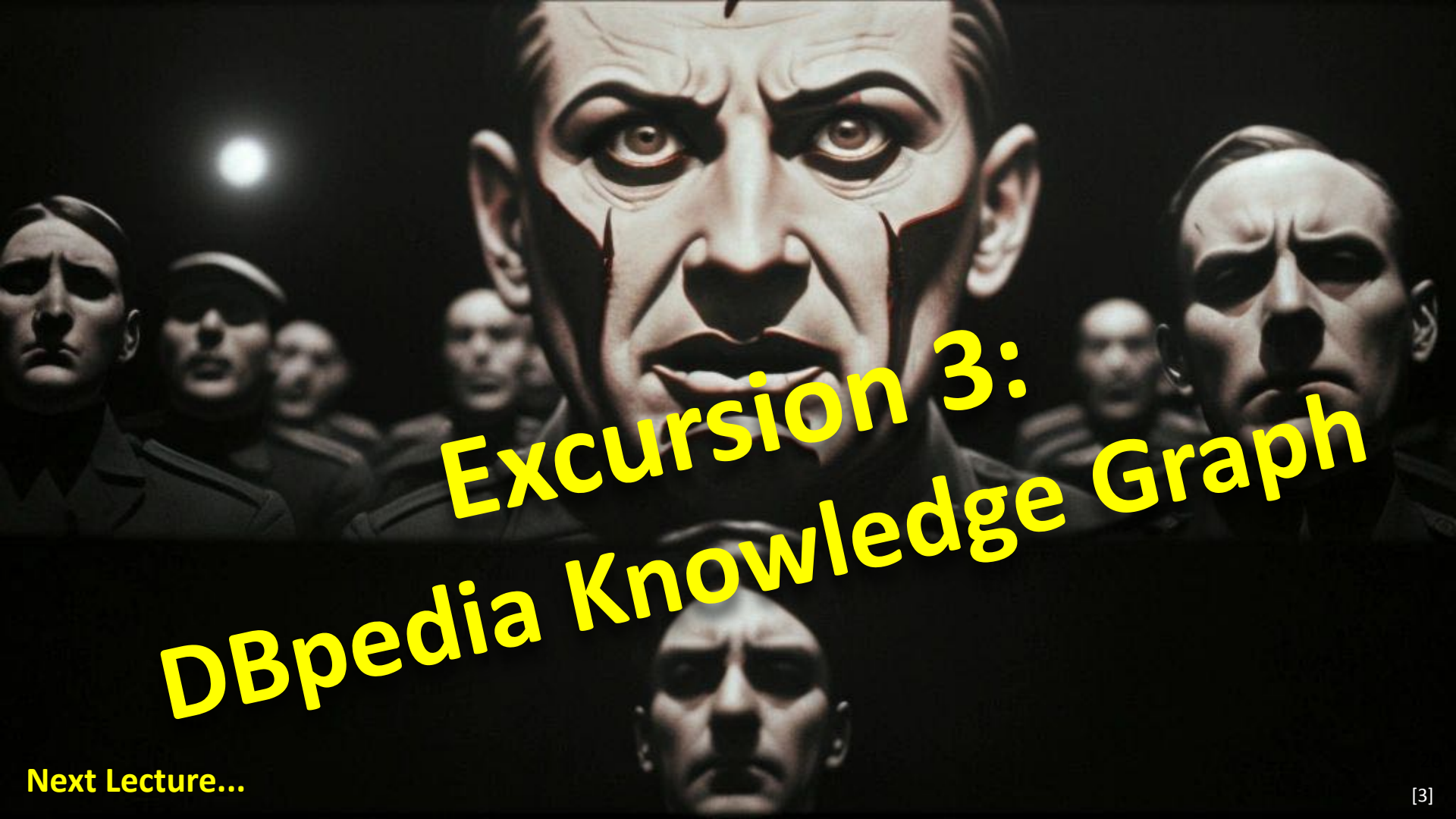
SPARQL Filter Constraints

```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
PREFIX dct:  <http://purl.org/dc/terms/>
PREFIX dbc:  <http://dbpedia.org/resource/Category:>

SELECT ?author_name ?title
FROM <http://dbpedia.org/>
WHERE {
       ?author rdf:type dbo:Writer .
       ?author rdfs:label ?author_name
       FILTER (LANG(?author_name)="en").
       ?work dbo:author ?author .
             ?work rdfs:label ?title .
       FILTER (LANG(?title)="en")
        ?work dct:subject dbc:Dystopian_novel .
} LIMIT 100
```

Example:
Search for **authors** and their **books**, filter results for **English labels** and **Dystopian novels** and **limit** the results to **the first 100**

query SPARQL endpoint



Next Lecture...

Bibliographic References:

- Steve Harris, Garlik, Andy Seaborne (2013), *SPARQL 1.1 Query Language*, W3C Recommendation 21 March 2013
- Aidan Hogan (2020), *The Web of Data*, Springer.
  Chap. 6.2.1 Basic Graph Patterns, pp. 628–633.

Picture References:

[1] "A dystopian city street scene clearly exhibiting the consequences of both unchecked population growth on society and the hoarding of resources by a wealthy minority in the style of a 1960s pulp cover.", created via ArtBot, Deliberate, 2023, [CC-BY-4.0], https://tinybots.net/artbot

[2] Benjamin Nowack, *The Semantic Web - Not a Piece of cake…*, at bnode.org, 2009-07-08 , [CC BY 3.0], https://web.archive.org/web/20220628120341/http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake

[3] "The close-up of a frightening fascist face looking from a large screen hanging over a huge audience of uniformed workers sitting in the dark movie theater in front of the screen watching in awe and fear, in the style of a 1930 expressionistic movie." , created via ArtBot, Deliberate 2023, [CC-BY-4.0], https://tinybots.net/artbot

[4] "In this picture in the style of a neofuturistic science fiction cover, an old man is sitting in a wheelchair is looking up high in the sky holding a small book in his hands. In the background there are superfuturistic lonely sky scrapers, entire cities floating in the air, and two giant moons." , created via ArtBot, Deliberate 2023, [CC-BY-4.0], https://tinybots.net/artbot