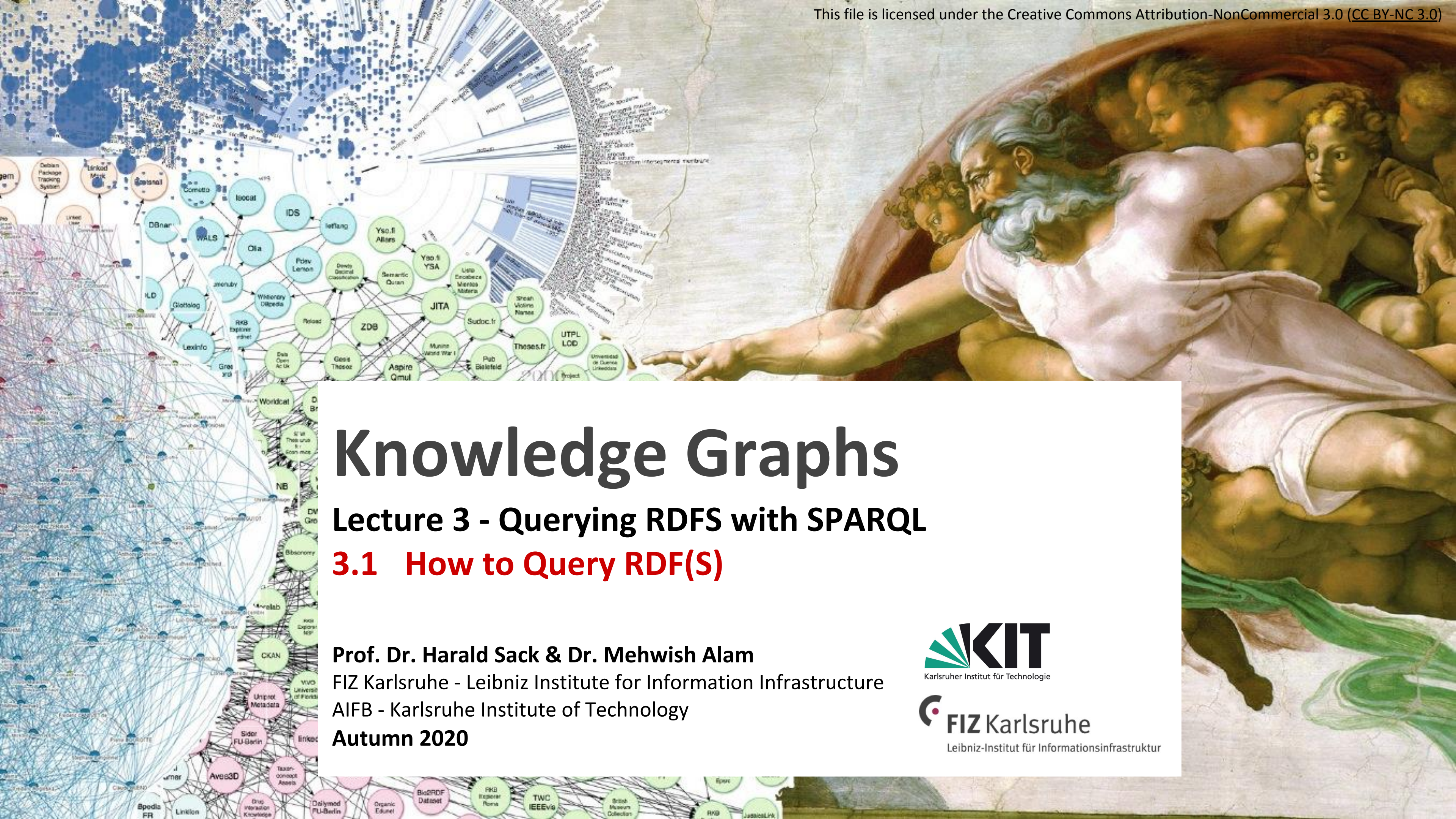
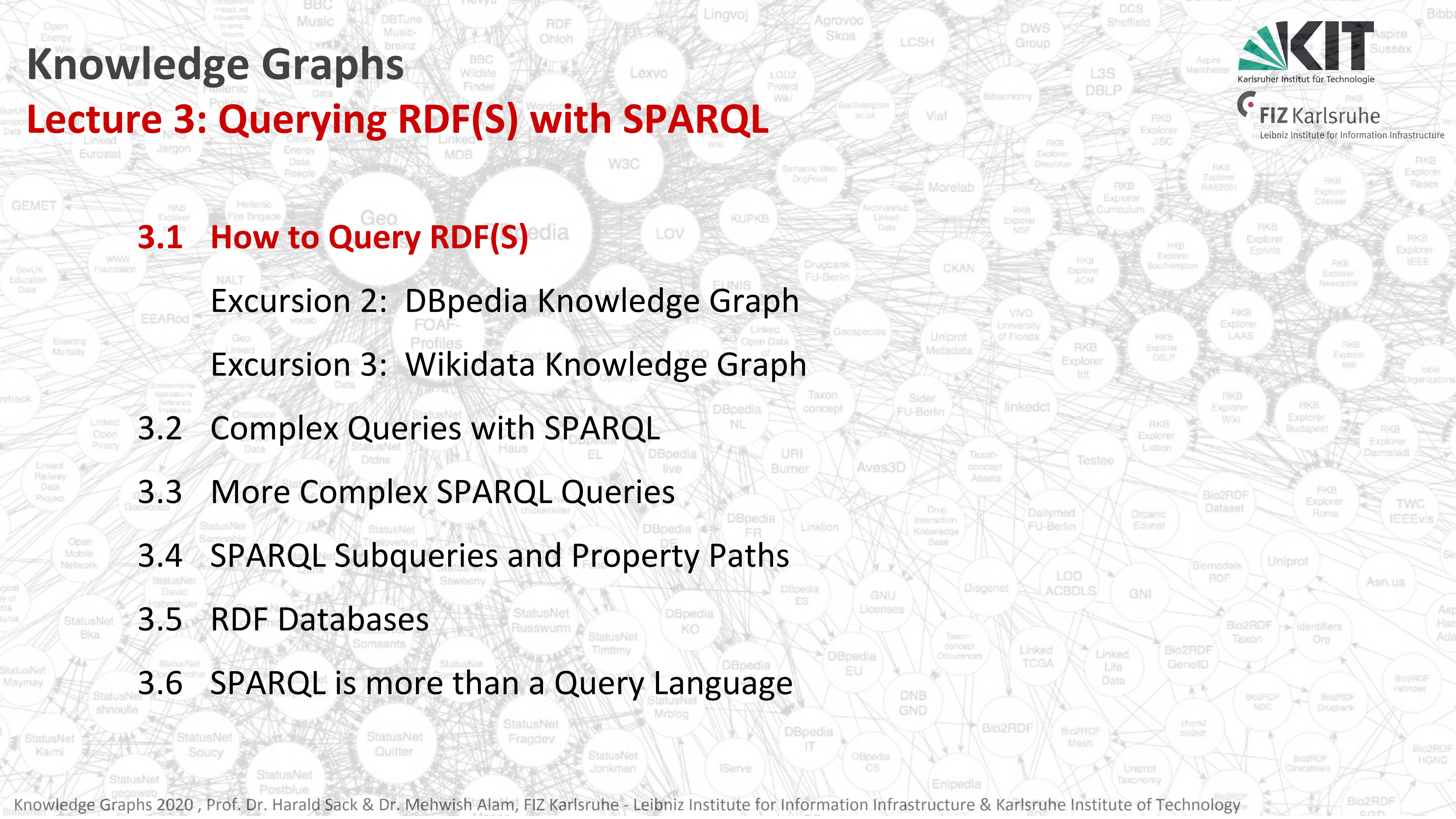


Knowledge Graphs

Lecture 3 - Querying RDFS with SPARQL

3.1 How to Query RDF(S)

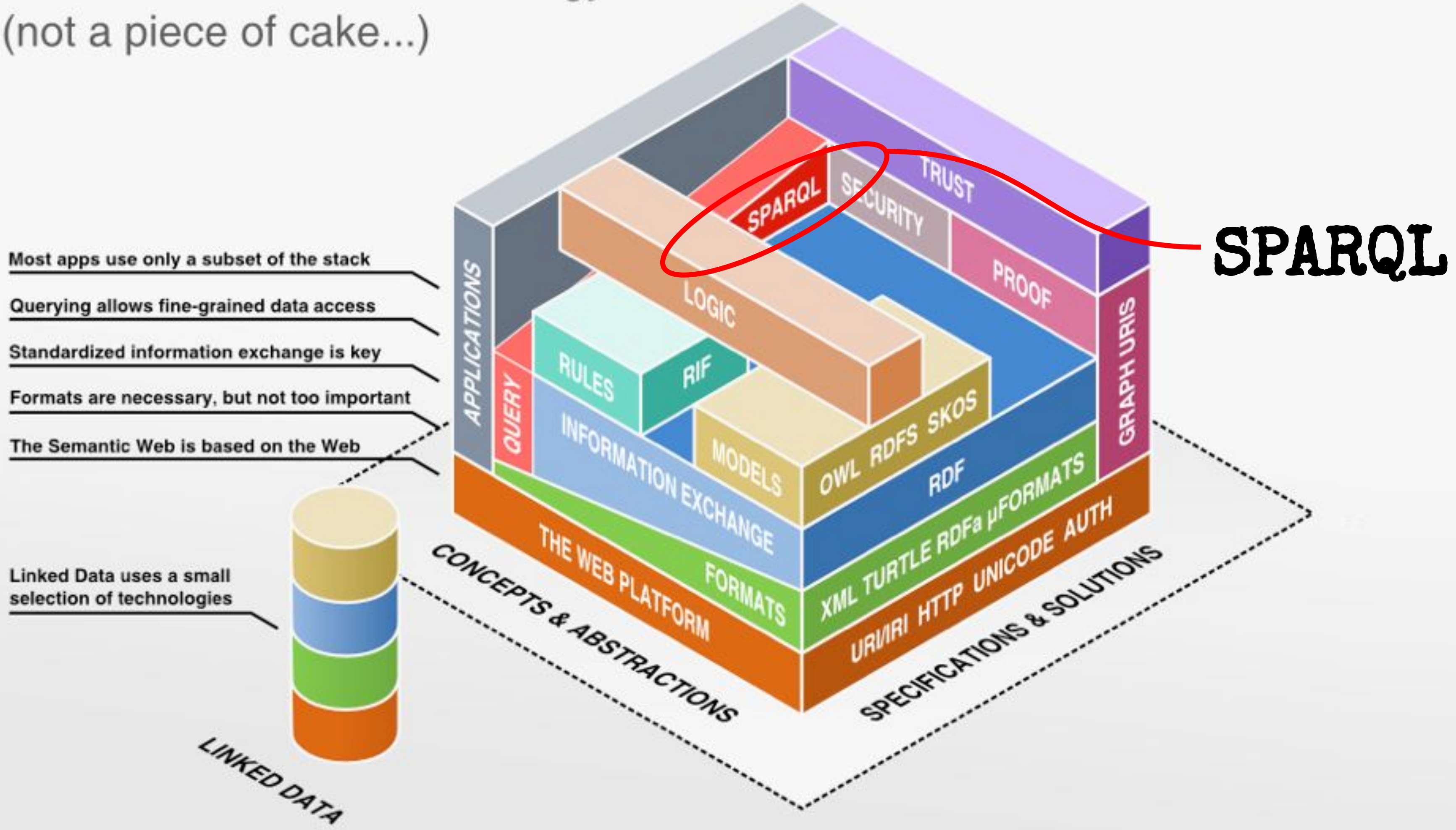
Prof. Dr. Harald Sack & Dr. Mehwish Alam
FIZ Karlsruhe - Leibniz Institute for Information Infrastructure
AIFB - Karlsruhe Institute of Technology
Autumn 2020



Knowledge Graphs

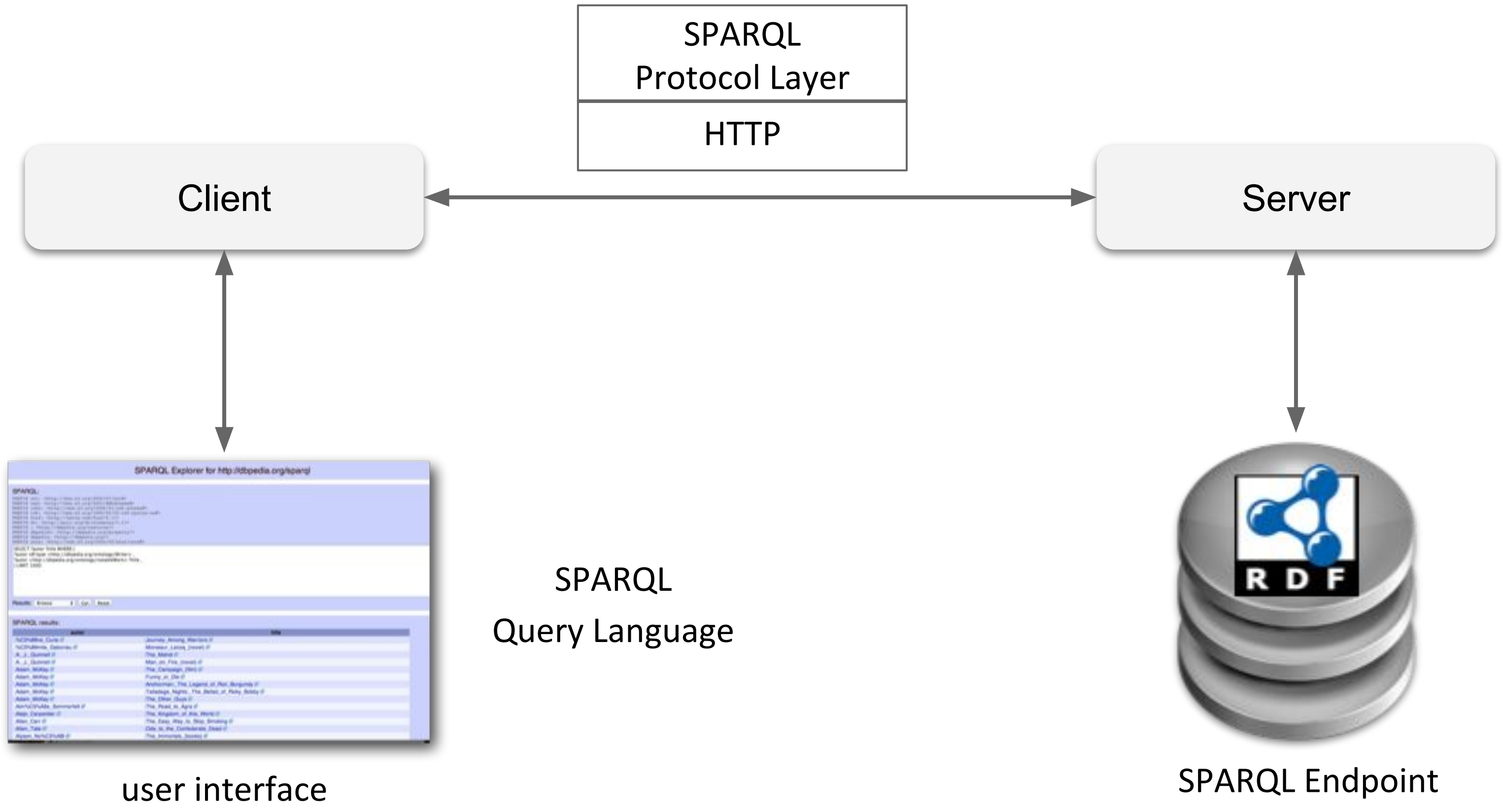
Lecture 3: Querying RDF(S) with SPARQL

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


Most apps use only a subset of the stack
Querying allows fine-grained data access
Standardized information exchange is key
Formats are necessary, but not too important
The Semantic Web is based on the Web
Linked Data uses a small selection of technologies
LINKED DATA
APPLICATIONS
QUERY
SPARQL
SECURITY
TRUST
PROOF
LOGIC
RULES
RIF
INFORMATION EXCHANGE
MODELS
OWL RDFS SKOS
RDF
GRAPH URIS
FORMATS
XML TURTLE RDFa µFORMATS
THE WEB PLATFORM
URI/IRI HTTP UNICODE AUTH
CONCEPTS & ABSTRACTIONS
SPECIFICATIONS & SOLUTIONS
SPARQL

[1]

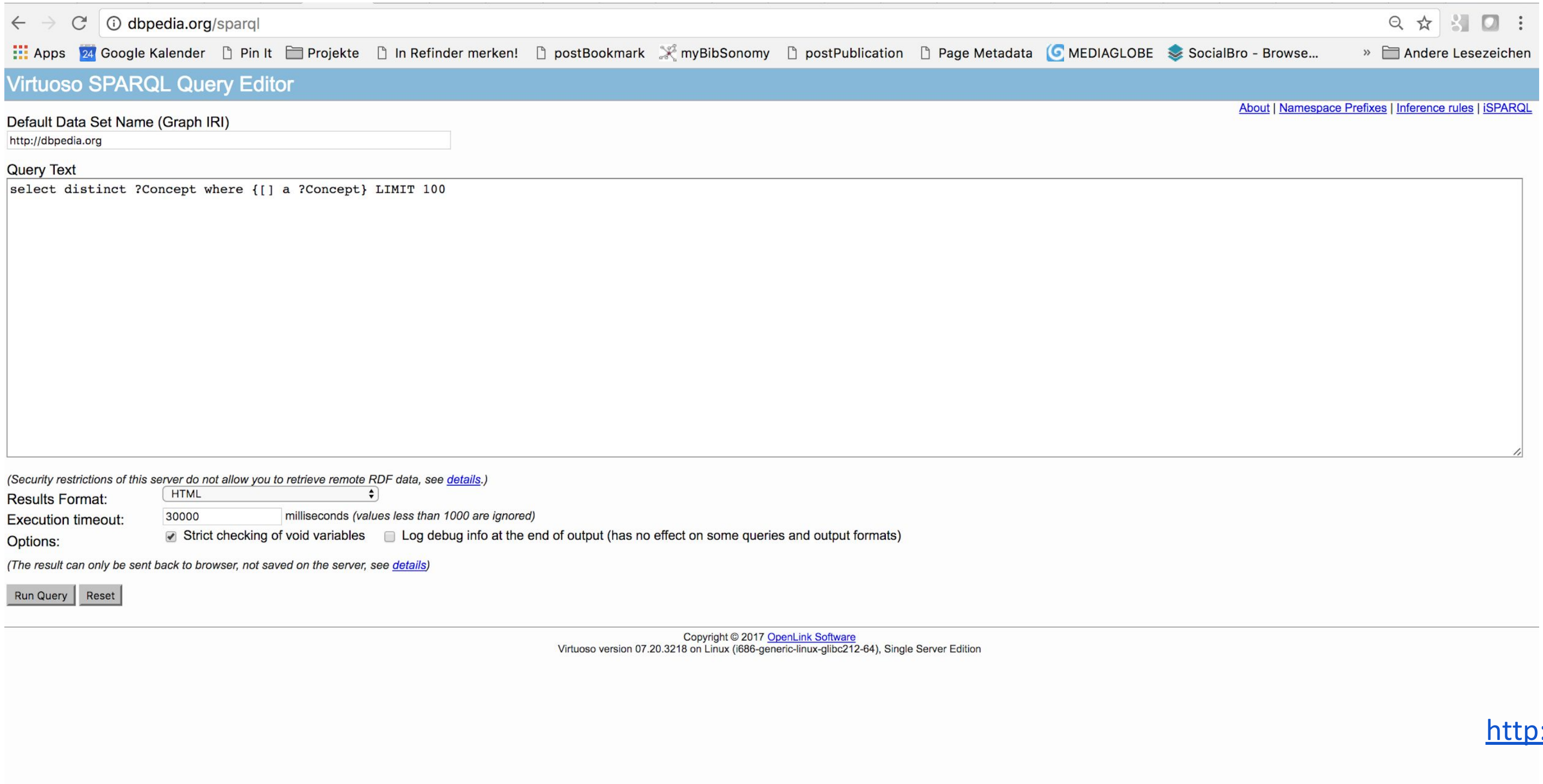
SPARQL - A Query Language for RDF(S)



SPARQL - A Query Language for RDF(S)

- **SPARQL P**rotocol **a**nd **R**DF **Q**uery **L**anguage is
  - a **Query Language** for RDF graph traversal (*SPARQL Query Language Specification*)
  - a **Protocol Layer**, to use SPARQL via http (*SPARQL Protocol for RDF Specification*)
  - an **XML Output Format Specification** for SPARQL queries (*SPARQL Query XML Results Format*)
  - W3C Standard (SPARQL 1.1, Mar 2013)
  - inspired by SQL

SPARQL - Endpoint Example


dbpedia.org/sparql
Apps Google Kalender Pin It Projekte In Refinder merken! postBookmark myBibSonomy postPublication Page Metadata MEDIAGLOBE SocialBro - Browse... Andere Lesezeichen
About | Namespace Prefixes | Inference rules | iSPARQL
Run Query
Reset

Virtuoso SPARQL Query Editor

Default Data Set Name (Graph IRI): http://dbpedia.org

Query Text

```
select distinct ?Concept where {[] a ?Concept} LIMIT 100
```

(Security restrictions of this server do not allow you to retrieve remote RDF data, see details.)

Results Format: HTML

Execution timeout: 30000 milliseconds *(values less than 1000 are ignored)*

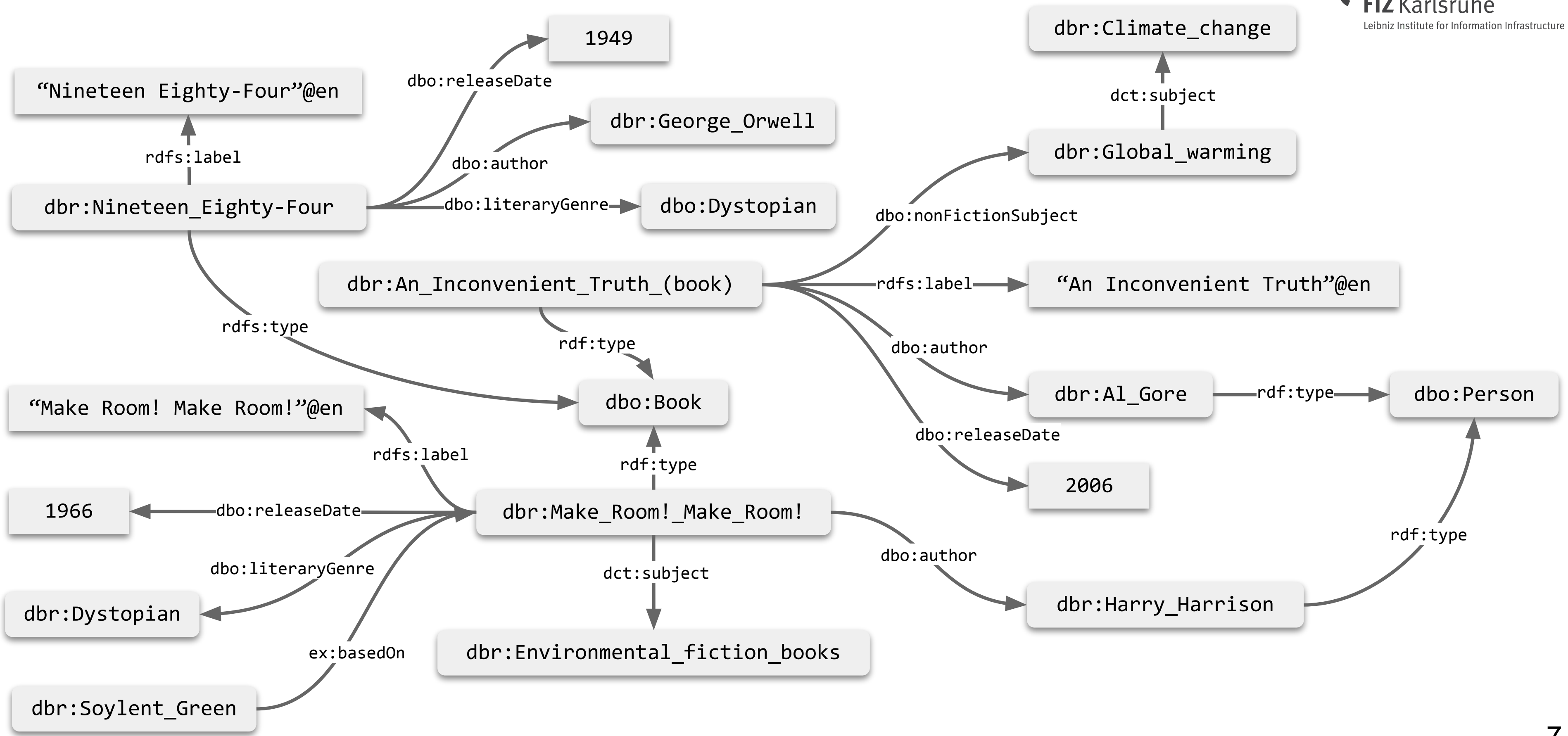
Options: [✓] Strict checking of void variables [ ] Log debug info at the end of output (has no effect on some queries and output formats)

(The result can only be sent back to browser, not saved on the server, see details)

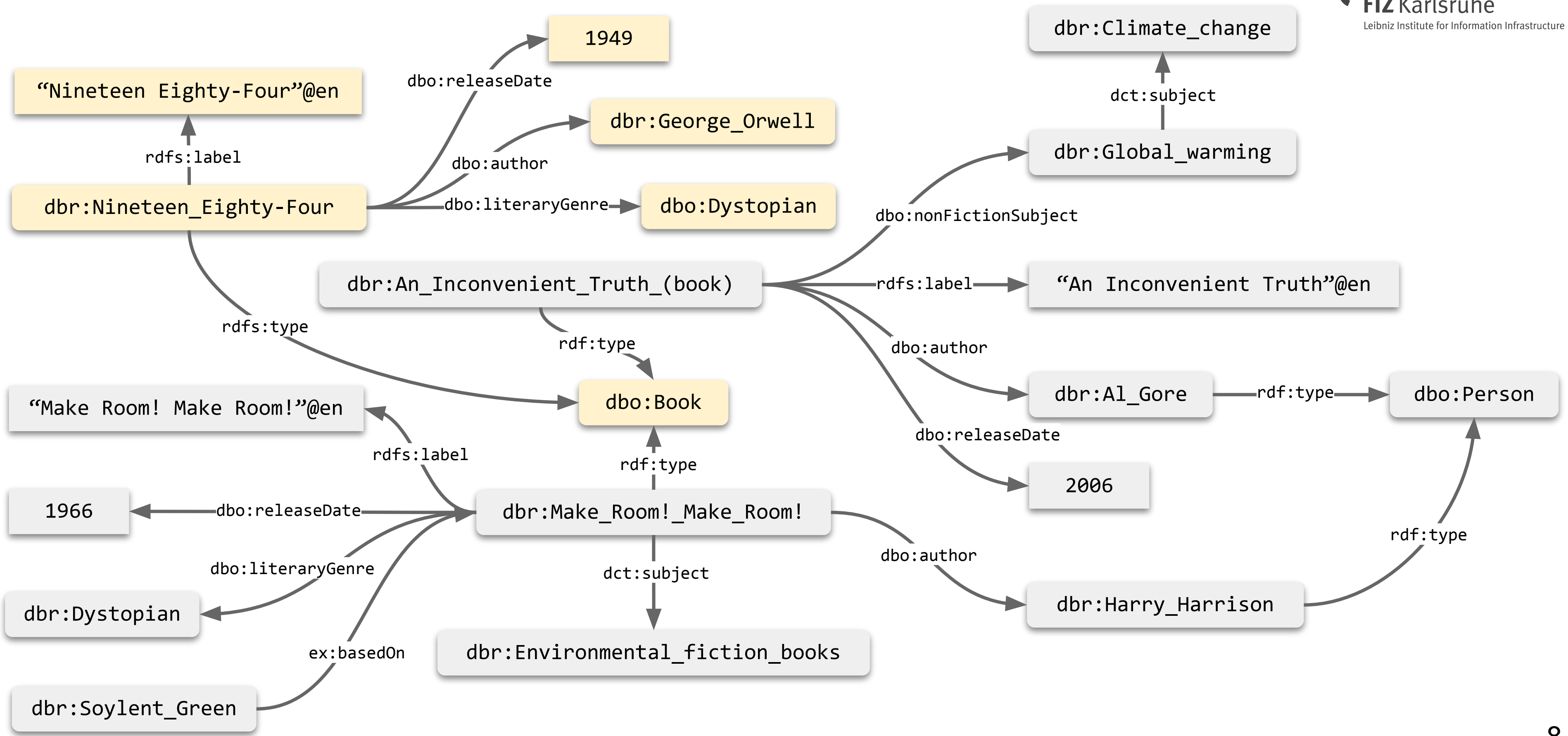
Copyright © 2017 OpenLink Software
Virtuoso version 07.20.3218 on Linux (i686-generic-linux-glibc212-64), Single Server Edition

http://dbpedia.org/sparql

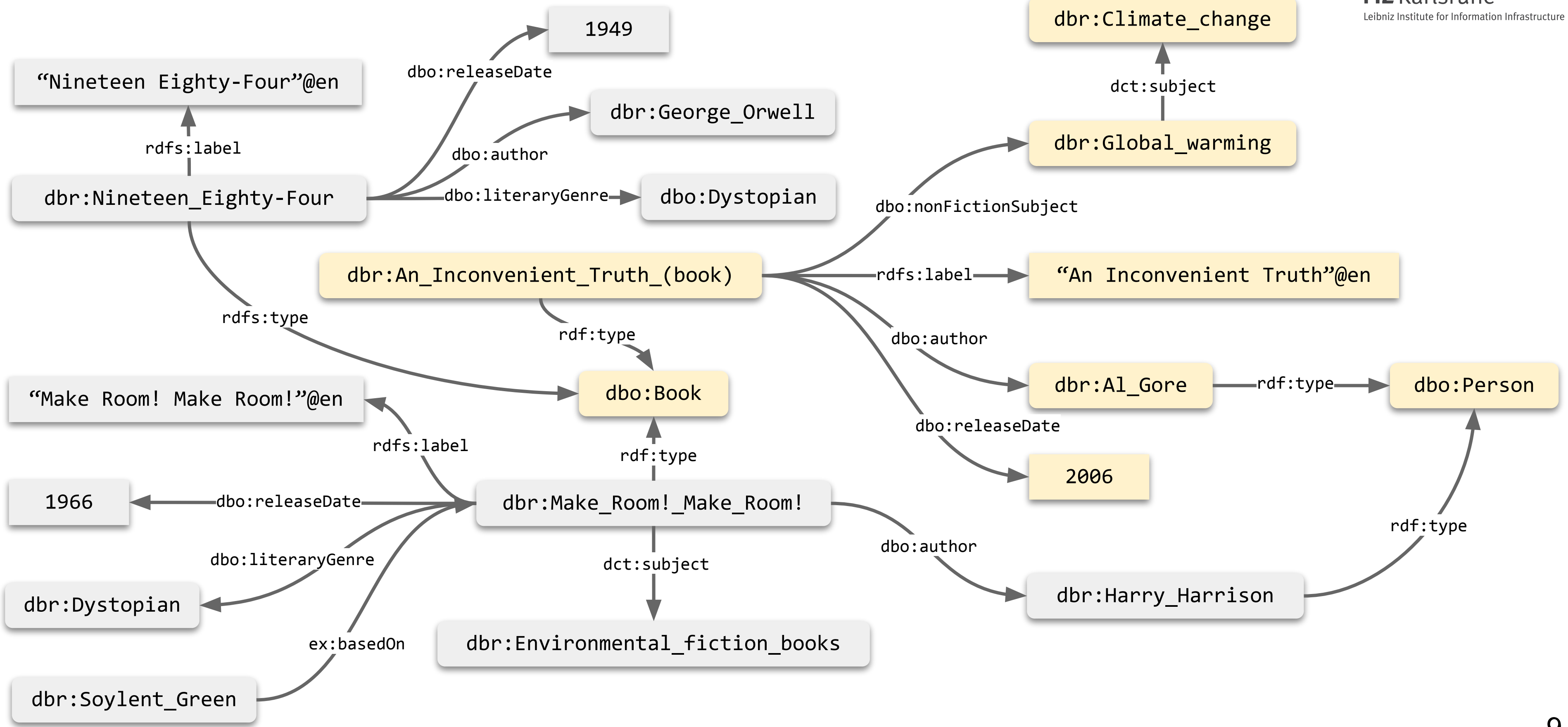
Querying an RDF-based Knowledge Graph



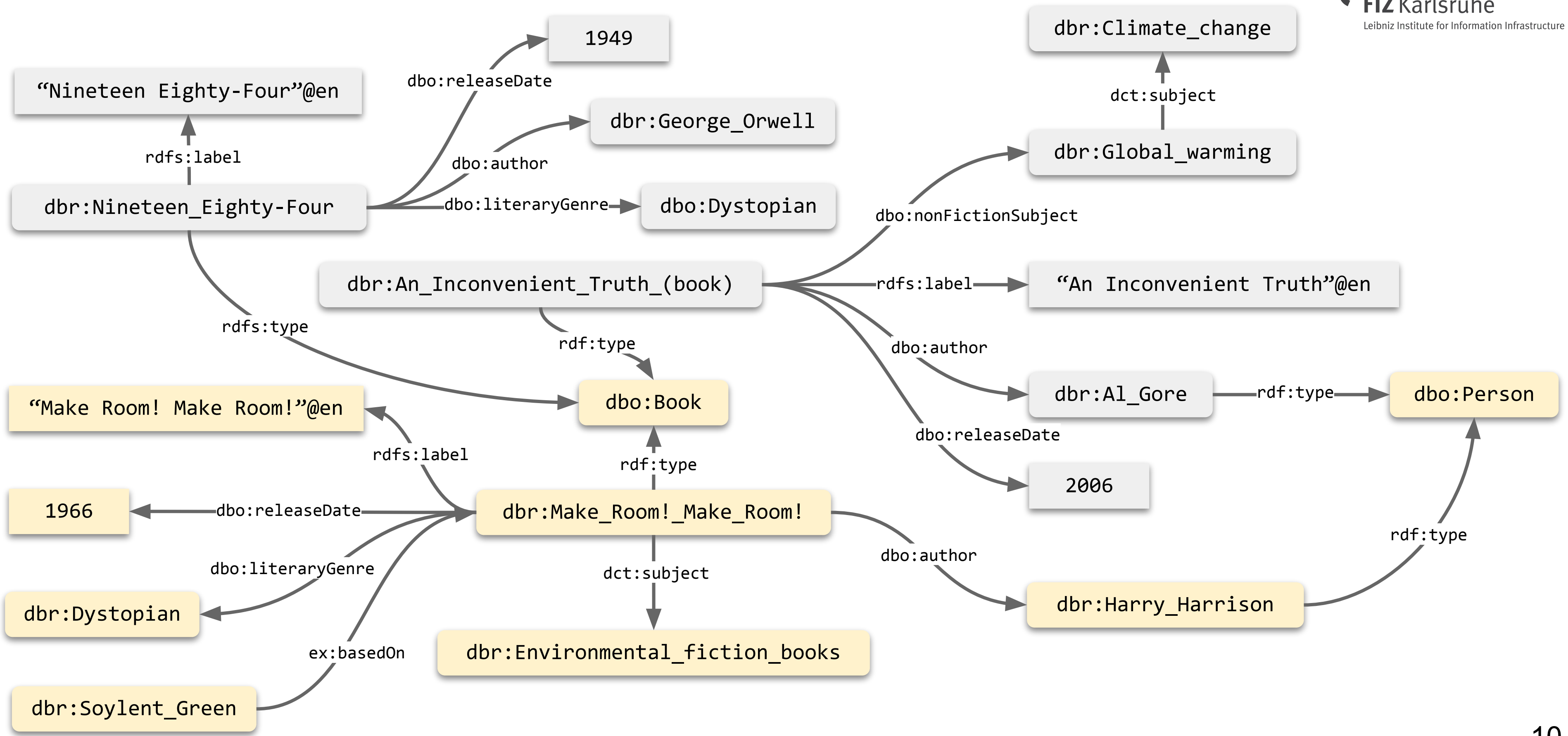
Querying an RDF-based Knowledge Graph



Querying an RDF-based Knowledge Graph



Querying an RDF-based Knowledge Graph



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
An Inconvenient Truth	Al Gore	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).

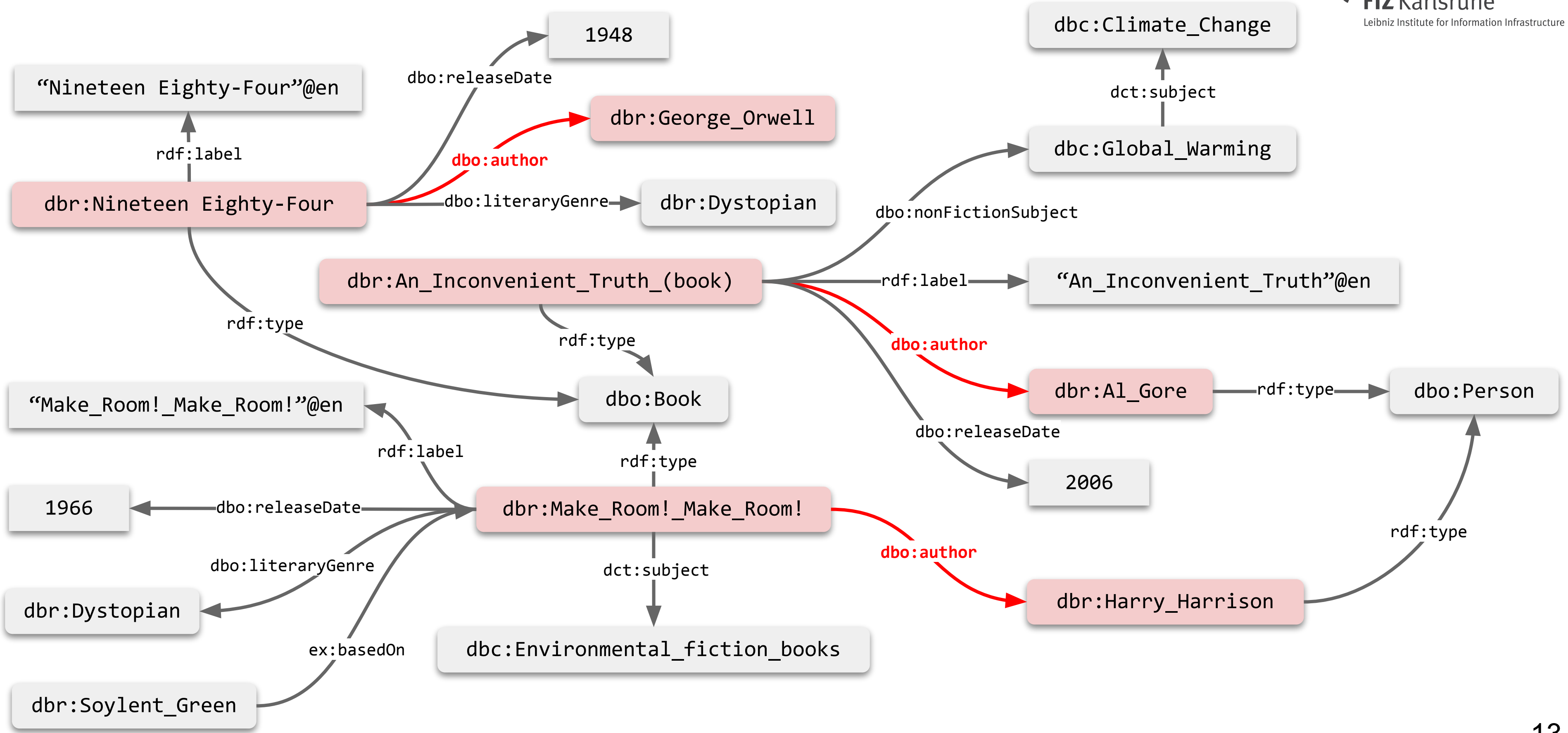
  Graph Pattern (Triple Pattern) = Turtle + Variables

- Example:
  Look for books and their authors (via property dbo:author):

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

variables

SPARQL Graph Pattern Matching



SPARQL Complex 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 Pattern Matching

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

- *Given a 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 Complex Pattern Matching

```
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 Complex Pattern Matching

```
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 Operator Constraints

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

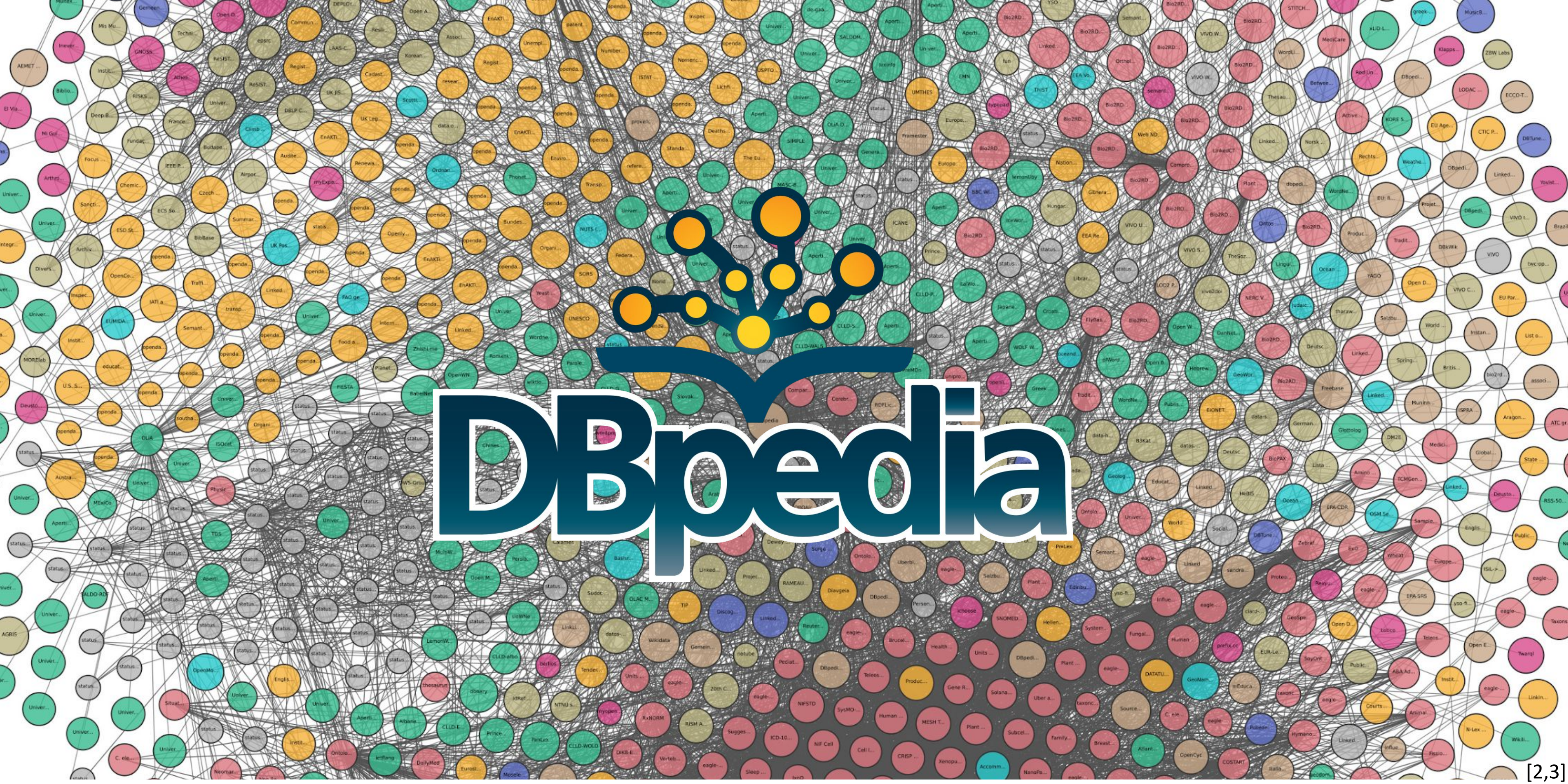
SPARQL Filter Constraints

```
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:Environmental_fiction_books .
} LIMIT 100
```

- Example:
  Search for **authors** and their **books**, filter results for **English labels** and **Environmental fiction books** and **limit** the results to **the first 100**

query SPARQL endpoint


[2,3]

Next Lecture: **Excursion 2: DBpedia Knowledge Graph**

Knowledge Graphs

3. Querying RDF(S) with SPARQL / 3.1 How to Query RDF(S)

Picture References:

- [1] Benjamin Nowack, *The Semantic Web - Not a Piece of cake…*, at bnode.org, 2009-07-08 , [CC BY 3.0] http://bnode.org/blog/2009/07/08/the-semantic-web-not-a-piece-of-cake
- [2] DBpedia logo, wiki.dbpedia.org, DBpedia Team [Public Domain] https://commons.wikimedia.org/wiki/File:DBpediaLogo.svg
- [3] The Linked Open Data Cloud, lod-cloud.net, [CC-BY] https://lod-cloud.net/clouds/lod-cloud.svg