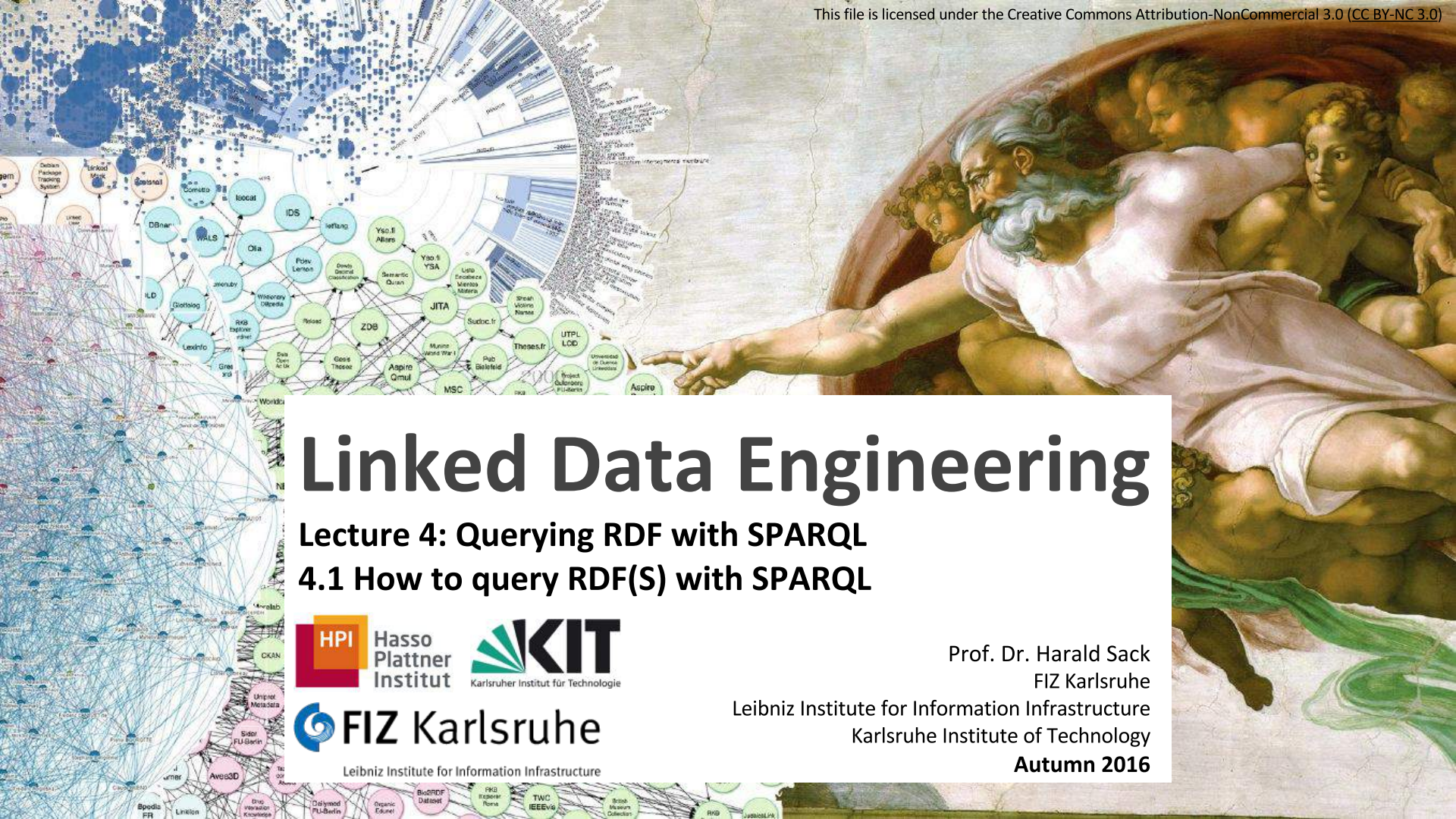


# Linked Data Engineering

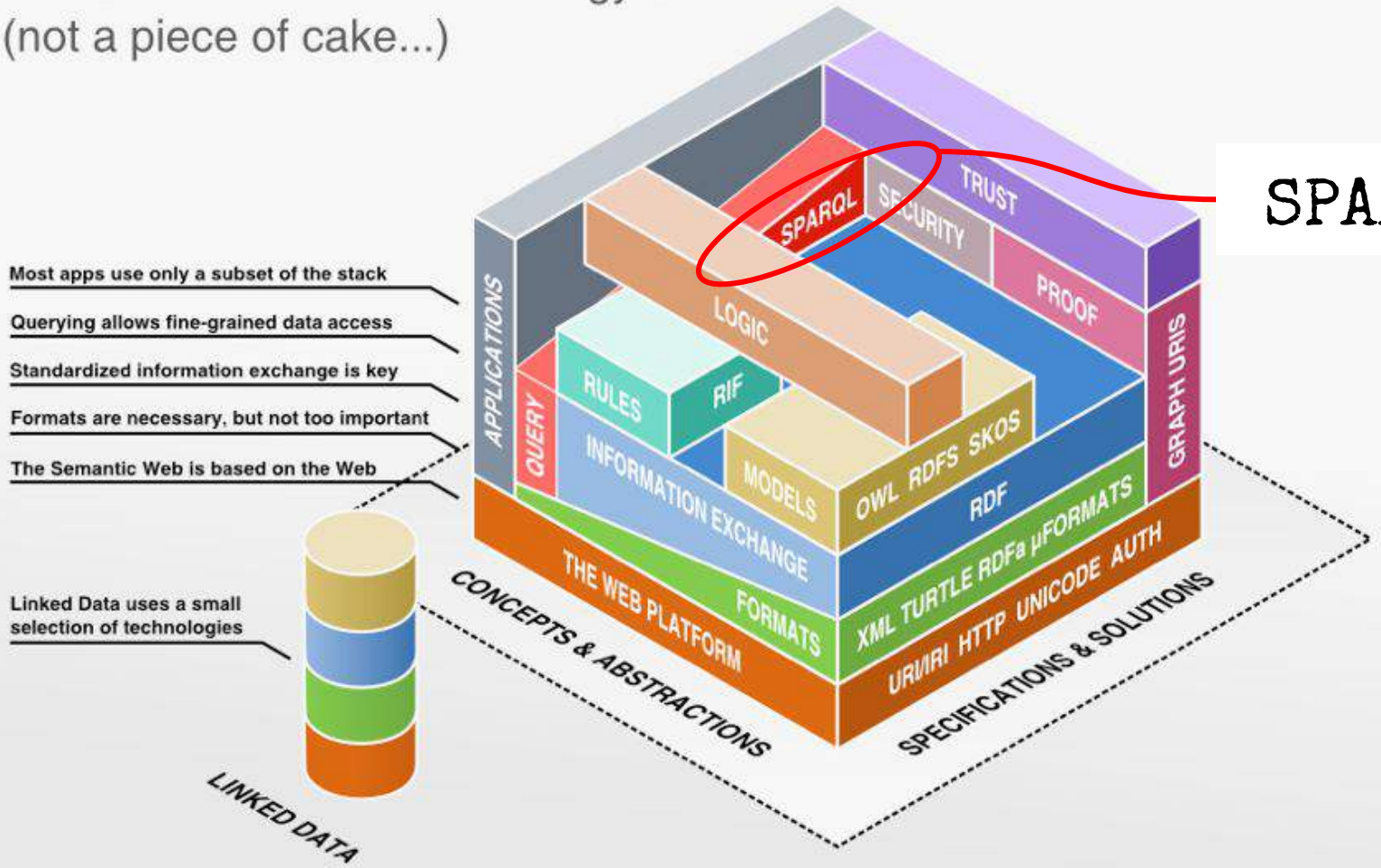
## Lecture 4: Querying RDF with SPARQL

## 4.1 How to query RDF(S) with SPARQL

Prof. Dr. Harald Sack
FIZ Karlsruhe
Leibniz Institute for Information Infrastructure
Karlsruhe Institute of Technology

**Autumn 2016**

# 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

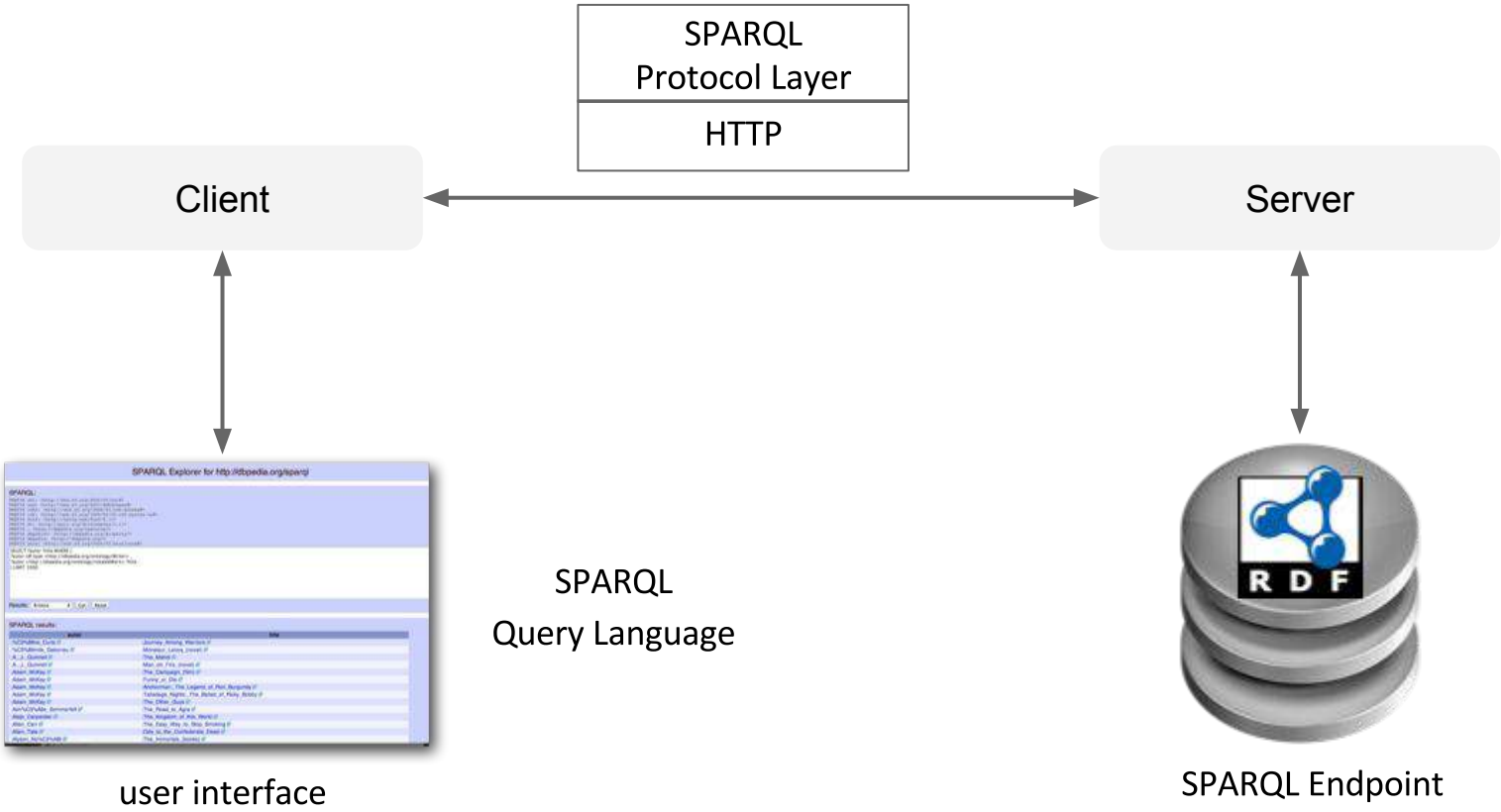
The Semantic Web is based on the Web

Linked Data uses a small selection of technologies

SPARQL

HPI Hasso Plattner Institut

# SPARQL - A Query Language for RDF



# SPARQL - A Query Language for RDF

- **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

# For Queries we need Variables

- SPARQL **Variables** are bound to RDF terms
  - e.g. **?title, ?author, ?address**
- In the same way as in SQL,
  a **Query for variables** is performed via **SELECT statement**
  - e.g. **SELECT ?title ?author ?published**

SPARQL Query

- A SELECT statement returns Query Results as a **table**

| ?title | ?author | ?published |
|---|---|---|
| 1984 | George Orwell | 1948 |
| Brave New World | Aldous Huxley | 1932 |
| Fahrenheid 451 | Ray Bradbury | 1953 |

SPARQL Result

# SPARQL - Graph Pattern Matching

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

  **(Graph) Triple Pattern = Turtle + Variables**

- Example:
  *Look for countries and their capitals:*

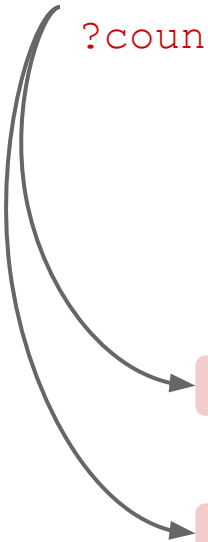
```
?country dbo:capital ?capital .
```

- A **Basic Graph Pattern** (**BGP**) is a set of Triple Pattern

# SPARQL - Graph Pattern Matching

Triple Pattern

```
?country dbo:capital ?capital .
```



RDF Graph

```
dbpedia:Venezuela rdf:type dbo:Country .
dbpedia:Venezuela dbo:capital dbpedia:Caracas .
dbpedia:Venezuela dbprop:language "Spanish" .
dbpedia:Germany rdf:type dbo:Country .
dbpedia:Germany dbo:capital "Berlin" .
dbpedia:Germany dbp:language "German" .
...
```

# SPARQL - Complex Query Patterns

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

- *Find countries, their capitals, and their population count:*

```
?country dbo:capital ?capital .
?country dbo:population ?population .
```

- *Given a FOAF URI, find the name of a person and her friends:*

```
<http://hpi-web.de/id#haraldsack> foaf:name  ?surname ;
                                  foaf:knows  ?friend .
?friend foaf:name ?friend_surname .
```

# SPARQL - General Query Format

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

```
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 .
}
```

*specifies namespaces*

*specifies output variables*

*specifies graph to be queried*

*specifies graph pattern to be matched*

# SPARQL - General Query Format

- *search 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:*

```
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*

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*

- `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

- Example: Filter results only for English labels

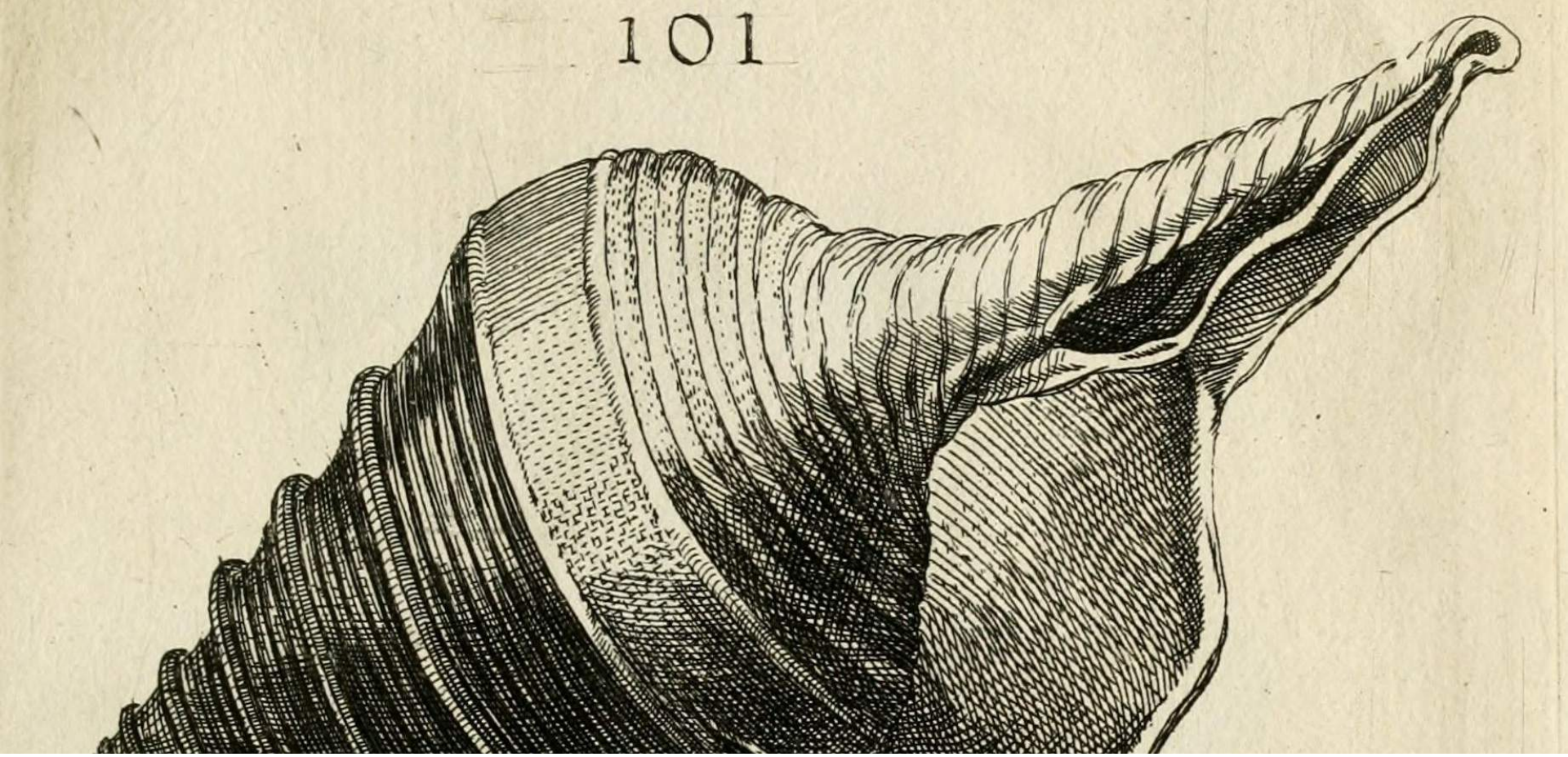
```
PREFIX : <http://dbpedia.org/resource/>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX dbo: <http://dbpedia.org/ontology/>
SELECT ?author_name ?title ?pages
FROM <http://dbpedia.org/>
WHERE {
       ?author rdf:type dbo:Writer .
       ?author rdfs:label ?author_name
       FILTER (LANG(?author_name)="en").
       ?author dbo:notableWork ?work .
       ?work dbo:numberOfPages ?pages .
       FILTER (?pages > 500)
       ?work rdfs:label ?title .
       FILTER (LANG(?title)="en")
} LIMIT 100
```



query SPARQL endpoint

# SPARQL - First Hands On

- From Wikipedia to DBpedia
  e.g. from http://en.wikipedia.org/wiki/George_Orwell to
  http://dbpedia.org/page/George_Orwell
- Browsing DBpedia
  e.g. using http://dbpedia.org/page/George_Orwell as a starting point to learn more about DBpedia structure and DBpedia ontologies
- Using DBpedia Sparql Endpoint with
  http://dbpedia.org/sparql
  and query DBpedia via SPARQL




**Next: 02 - Complex Queries with SPARQL**

Lecture 4 - Querying RDF with SPARQL - OpenHPI - Course Linked Data Engineering