What good is a Web of Linked Data if we cannot reliably query it?
MORE THAN HALF of public SPARQL endpoints availability < 95%
WE CANNOT QUERY public Linked Data reliably.

WE CANNOT BUILD applications on top of public queryable data.
It’s not a performance issue, it is an architectural problem.
An architectural problem requires an architectural solution.
We developed an approach to query Linked Data in a scalable and reliable way by moving intelligence from the server to the client.
What **Linked Data Fragments** are.

How clients can execute **queries**.

Taking querying to the **next level**.
What **Linked Data Fragments** are.

How clients can execute queries.

Taking querying to the next level.
Currently, there are three ways to query a Linked Data set.

They offer **fragments** of a dataset.
Any fragment of a Linked Data set is called a **Linked Data Fragment**.
Can we query fragments that balance client and server effort?

- High client effort
  - Data dump
  - Dereferencing
  - All
- Basic Linked Data Fragments
  - Subject
  - Triple pattern
- High server effort
  - SPARQL endpoint
  - SPARQL query
A *basic* LDF is easy to generate yet enables efficient querying.

**data** (*in pages*)

basic triple pattern \{ ?s ?p ?o. \}

**metadata**

count of total matches

**controls**

retrieve other basic LDFs
Linked Data Fragments Server

Data source dbpedia

Query dbpedia-virtuoso by triple pattern

subject: 
predicate: rdf:type
object: dbpedia-owl:Artist

Find matching triples

Total matches: ±61,073

metadata (total count)

data (first 100)

controls (other basic LDFs)
How can a server **publish**
basic Linked Data Fragments?

**open-source server**

choose your **back-end**

(*private*) SPARQL endpoint

HDT binary triple format

Turtle file

...
Web-Scale Querying through Linked Data Fragments

What Linked Data Fragments are.

How clients can execute queries.

Taking querying to the next level.
How to answer this query using only basic Linked Data Fragments?

```
SELECT ?person ?city WHERE {
  ?person a dbpedia-owl:Artist.
  ?city foaf:name "York"@en.
}
```
Get the corresponding fragments

?person a dbpedia-owl:Artist.
  dbpedia:Aamir_Zaki a dbpedia-owl:Artist.
  dbpedia:Ahmad_Morid a dbpedia-owl:Artist.
  ...

  dbpedia:Ganesh_Ghosh ...:birthPlace dbpedia:Bengal_Presidency.
  dbpedia:Jacques_L'enfant ...:birthPlace dbpedia:Beauce.
  ...

?city foaf:name "York"@en.
  ...
Get the corresponding fragments and read the **count** metadata.

```
?person a dbpedia-owl:Artist. ±61,000  
dbpedia:Aamir_Zaki a dbpedia-owl:Artist.  
dbpedia:Ahmad_Morid a dbpedia-owl:Artist. 
...

?person dbpedia-owl:birthPlace ?city. ±470,000  
dbpedia:Ganesh_Ghosh ...:birthPlace dbpedia:Bengal_Presidency.  
dbpedia:Jacques_L'enfant ...:birthPlace dbpedia:Beauce.  
...

?city foaf:name "York"@en. 12  
...
```
Start with the **smallest fragment.**
Start with the **first match.**

```plaintext
?person a dbpedia-owl:Artist ±61,
  dbpedia:Aamir_Zaki
  dbpedia:Ahmad_Morid a dbpedia-owl:Artist.
...

?person dbpedia-owl:birthPlace ±470,
  dbpedia:Ganesh_Ghosh ...:birthPlace dbpedia:Bengal_Presidency.
  dbpedia:Jacques_L'enfant ...:birthPlace dbpedia:Beauce.
...

?city foaf:name "York"@en. 12
...
```
How to answer this query using only basic Linked Data Fragments?

```sql
SELECT   ?person
WHERE {
    ?person   a   dbpedia-owl:Artist.
    dbpedia:York   foaf:name   "York"@en.
}
```
Get the corresponding fragments

?person a dbpedia-owl:Artist.
  dbpedia:Aamir_Zaki a dbpedia-owl:Artist.
  dbpedia:Ahmad_Morid a dbpedia-owl:Artist.
...

  dbpedia:John_Flaxman dbo:birthPlace dbpedia:York.
...

Get the corresponding fragments and read the **count** metadata.

\[ ?\text{person} \ a \ dbpedia\text{-}owl:\text{Artist}. \pm 61,000 \]

\[ dbpedia:\text{Aamir} \_\text{Zaki} \ a \ dbpedia\text{-}owl:\text{Artist}. \]
\[ dbpedia:\text{Ahmad} \_\text{Morid} \ a \ dbpedia\text{-}owl:\text{Artist}. \]

...

\[ ?\text{person} \ dbpo:\text{birthPlace} \ dbpedia:\text{York}. \ 75 \]

\[ dbpedia:\text{John} \_\text{Flaxman} \ dbpo:\text{birthPlace} \ dbpedia:\text{York}. \]
\[ dbpedia:\text{Joseph} \_\text{Hansom} \ dbpo:\text{birthPlace} \ dbpedia:\text{York}. \]

...
Start with the smallest fragment. Start with the first match.

?person a dbpedia-owl:Artist
  dbpedia:Aamir_Zaki
  dbpedia:Ahmad_Morid a dbpedia-owl:Artist.
...

  dbpedia:John_Flaxman dbo:birthPlace dbpedia:York.
...

How to answer this query using only basic Linked Data Fragments?

ASK {
    dbp:John_Flaxman a dbpo:Artist.
    dbp:John_Flaxman dbpo:birthPlace dbp:York.
    dbp:York foaf:name "York"@en.
}
Get the corresponding fragment and read the **count** metadata.

\[ \text{dbpedia:John_Flaxman} \ a \ \text{dbpedia-owl:Artist.} \quad 1 \]

\[ \text{dbpedia:John_Flaxman} \ a \ \text{dbpedia-owl:Artist.} \]

Output the match:

\[ ?\text{person} = \text{dbpedia:John_Flaxman} \]
\[ ?\text{city} = \text{dbpedia:York} \]
Recursively **repeat** the process for all bindings.

```sparql
  dbpedia:John_Flaxman dbo:birthPlace dbpedia:York.
...

?city foaf:name "York"@en.
...
```
Linked Data Fragments client

Enter a basic graph pattern query below and see how your browser solves it using only Basic Linked Data Fragments.

Enter a SPARQL query ...or choose an example Artists from all places named “York”

```
SELECT ?p, ?c WHERE {
}
```

Answer using Web data data source DBpedia Basic Linked Data Fragments

Query results
Web-Scale Querying through Linked Data Fragments

What Linked Data Fragments are.

How clients can execute queries.

Taking querying to the next level.
Linked Data Fragments is a vision, not just a single technology.

How can clients query the Web in a scalable way?
We want to query different servers, with many different kinds of fragments.

Linked Data Fragments

data dump
dereferencing
basic Linked Data Fragments
SPARQL endpoint
Find suitcases on Amazon and their cost.

```
SELECT ?label ?cost WHERE {
  ?suitcase schema:keywords "suitcase";
  prov:wasDerivedFrom <http://amazon.com/>;
  rdfs:label ?label;
  schema:cost ?cost.
}
```
Find suitcases on Amazon and see how much they cost on eBay.

```
SELECT  ?label  ?costA  ?costE
WHERE {
  ?suitcaseA schema:keywords "suitcase";
  prov:wasDerivedFrom <http://amazon.com/>;
  rdfs:label ?label;
  schema:cost ?costA.
  ?suitcaseE schema:keywords ?label;
  prov:wasDerivedFrom <http://ebay.com/>;
  schema:cost ?costE.
}
```
The Linked Data Fragments vision allows clients to query the Web.

If we want to see intelligent clients, we must stop building intelligent servers.

Linked Data Fragments is the quest to design servers that enable clients to query.
All software is available as open source.

linkeddatafragments.org

data.linkeddatafragments.org

client.linkeddatafragments.org
linkeddatafragments.org

Ruben Verborgh
Sam Coppens
Miel Vander Sande
Erik Mannens
Pieter Colpaert
Rik Van de Walle

Ghent University – iMinds – Multimedia Lab