

# Effiziente Anfragebearbeitung auf verteilten RDF-Datenquellen

Katja Hose

Max-Planck-Institut für Informatik  
Saarbrücken, Germany

17. Januar 2012

# Semantic Web und Linked Open Data

## Semantic Web

- Web 1.0, Web 2.0, Web 3.0, Web ???
- Informationen für Maschinen “interpretierbar”
- URIs für Dokumente und Konzepte
- Standards: OWL, RDF, RDFS, RDFa, SPARQL, etc.

## Linked Open Data Sources

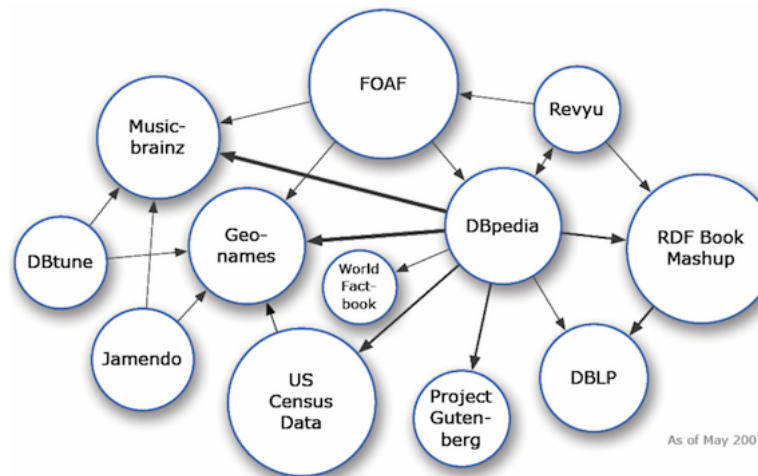
- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

# Semantic Web und Linked Open Data

## Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

Mai 2007

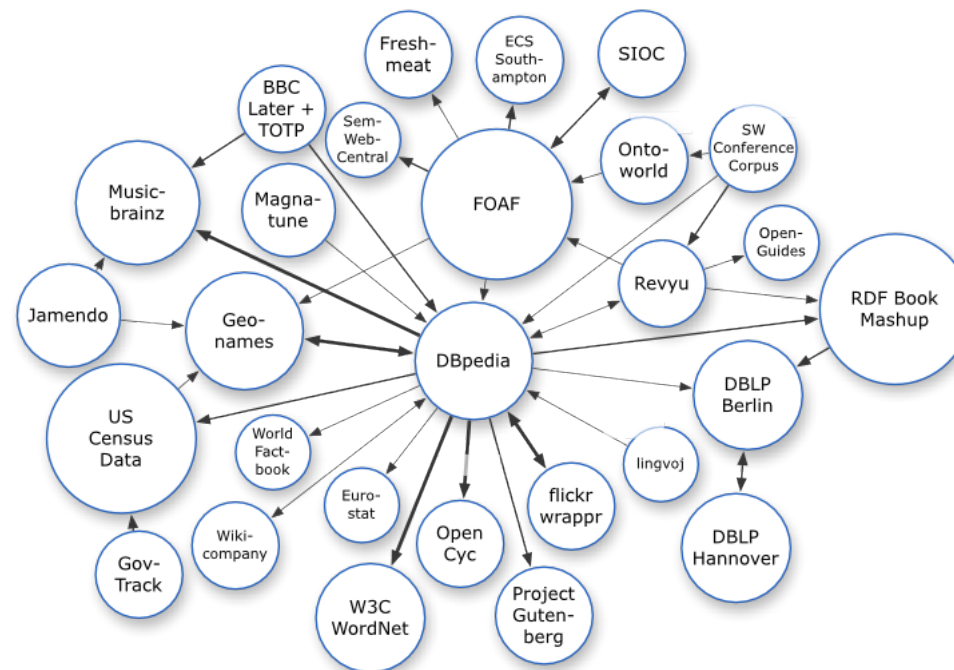


# Semantic Web und Linked Open Data

## Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

November 2007



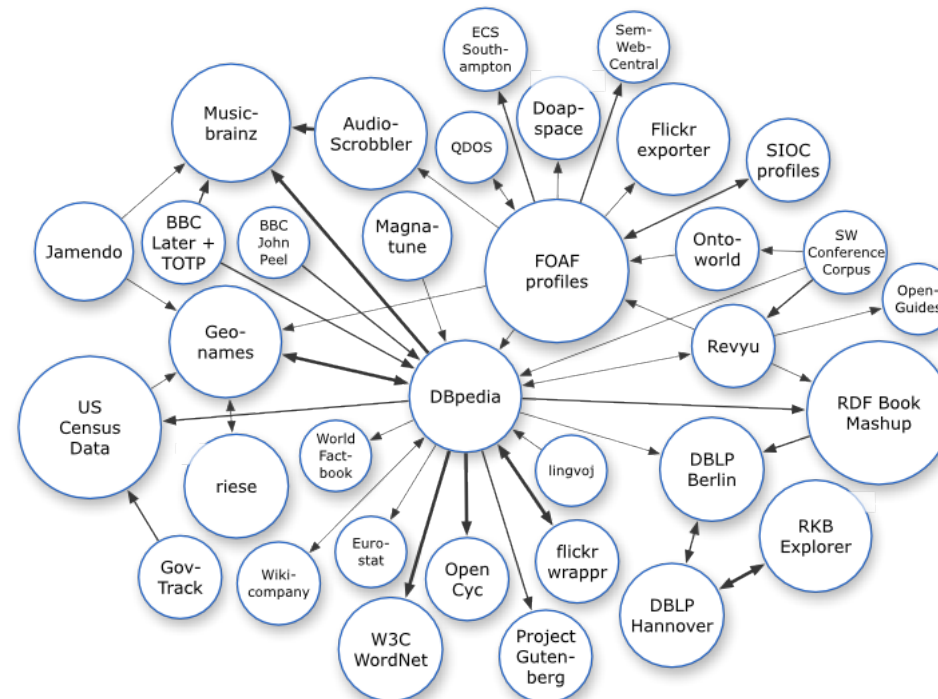


# Semantic Web und Linked Open Data

## Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

März 2008

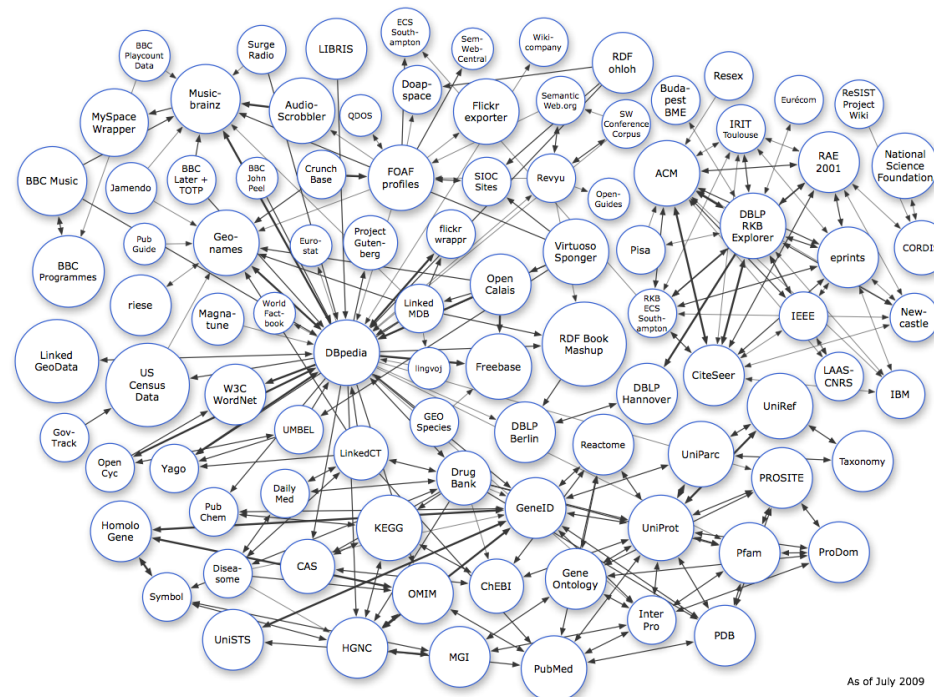


# Semantic Web und Linked Open Data

## Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

Juli 2009

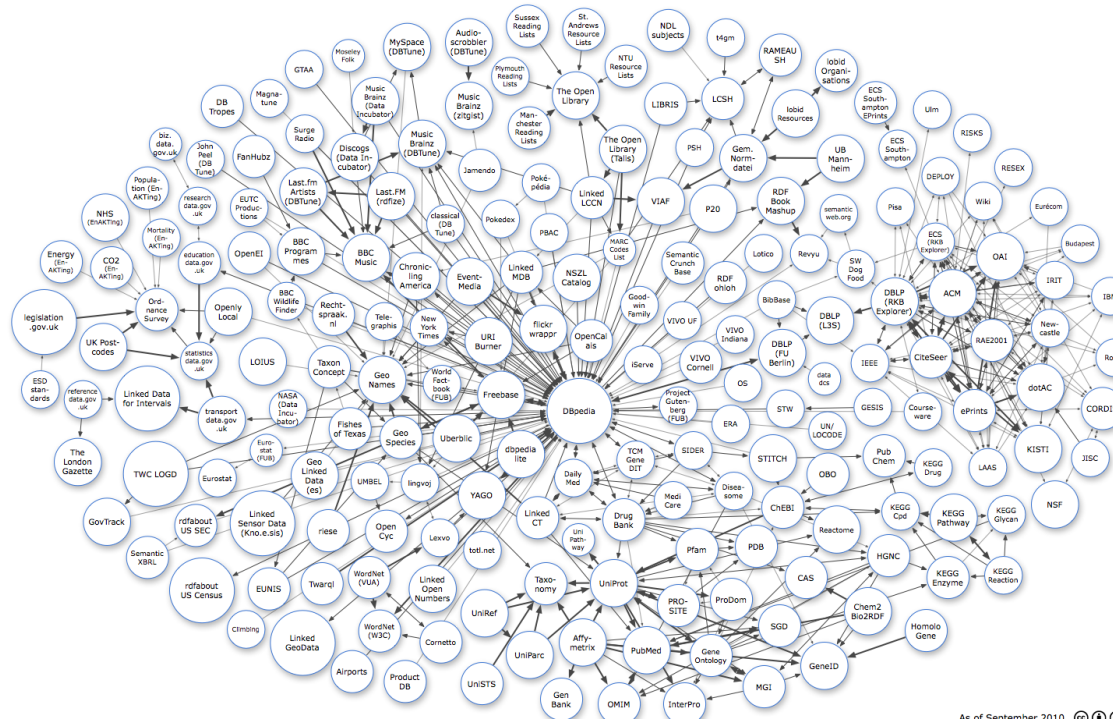


As of July 2009

# Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

# September 2010

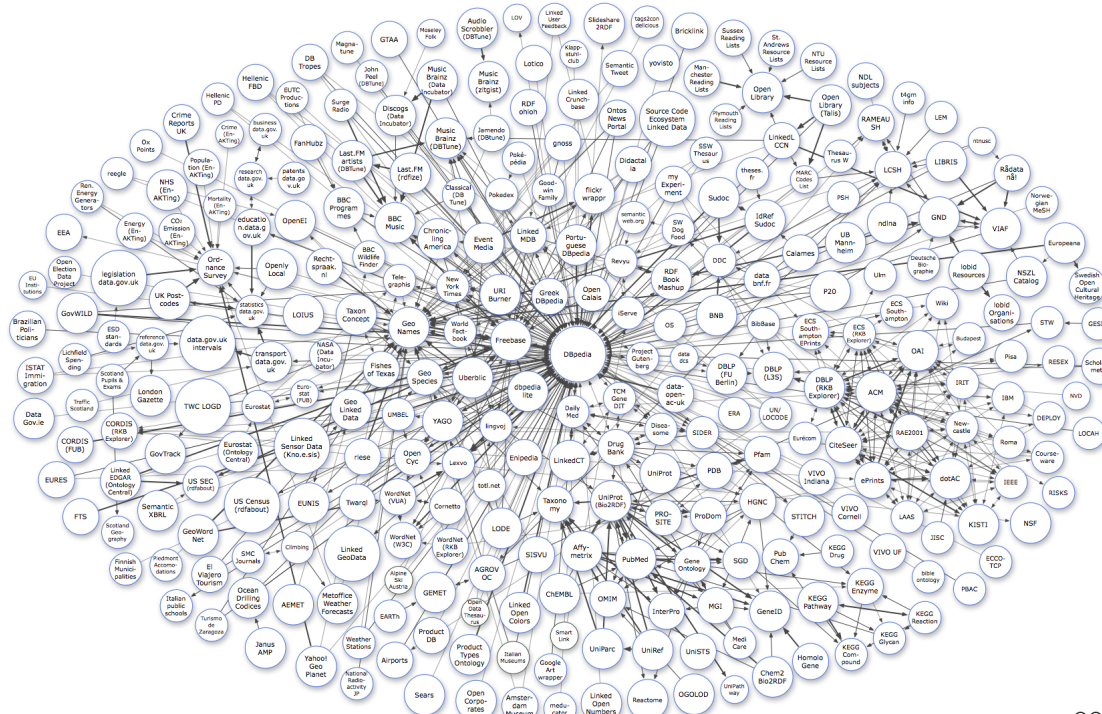


# Semantic Web und Linked Open Data

## Linked Open Data Sources

- DBpedia, YAGO, OpenCyc, DBLP, ACM, ...
- RDF, SPARQL
- Stetiges Wachstum
- Links/Referenzen zwischen Quellen

September 2011



As of September 2011 © 1 2

# Linked Open Data

## Linked Open Data (LOD)

- Daten liegen als RDF verteilt vor (Media, Geographic, Publications, User-Generated Content, Government, Cross-Domain, Life Sciences)
- Inhaltliche Überlappungen
  - Schauspieler, Filme: DBpedia, Yago, IMDB,...
  - Orte: Geonames, Freebase, NYTimes, DBpedia, Yago,...
- Links beschreiben Beziehungen (RDF Triple)  
`<http://www.imdb.com/name/nm0817881> owl:sameAs`  
`<http://dbpedia.org/resource/Carlo_Pedersoli>`

# Linked Open Data

## Design Issues<sup>1</sup>

- Use URIs as names for things
- Use HTTP URIs so that people can look up those names
- When someone looks up a URI, provide useful information, using the standards (RDF, SPARQL)
- Include links to other URIs, so that they can discover more things

---

<sup>1</sup><http://www.w3.org/DesignIssues/LinkedData.html>

# Datenquellen

Daten stehen auf unterschiedliche Weise zur Verfügung

- SPARQL Endpoints
- Dereferenzierbare URIs
- RDF Dumps
- Annotierte Webseiten
- Web Services
- ...

# Wer ist Carlo Pedersoli?

Linked Open Data

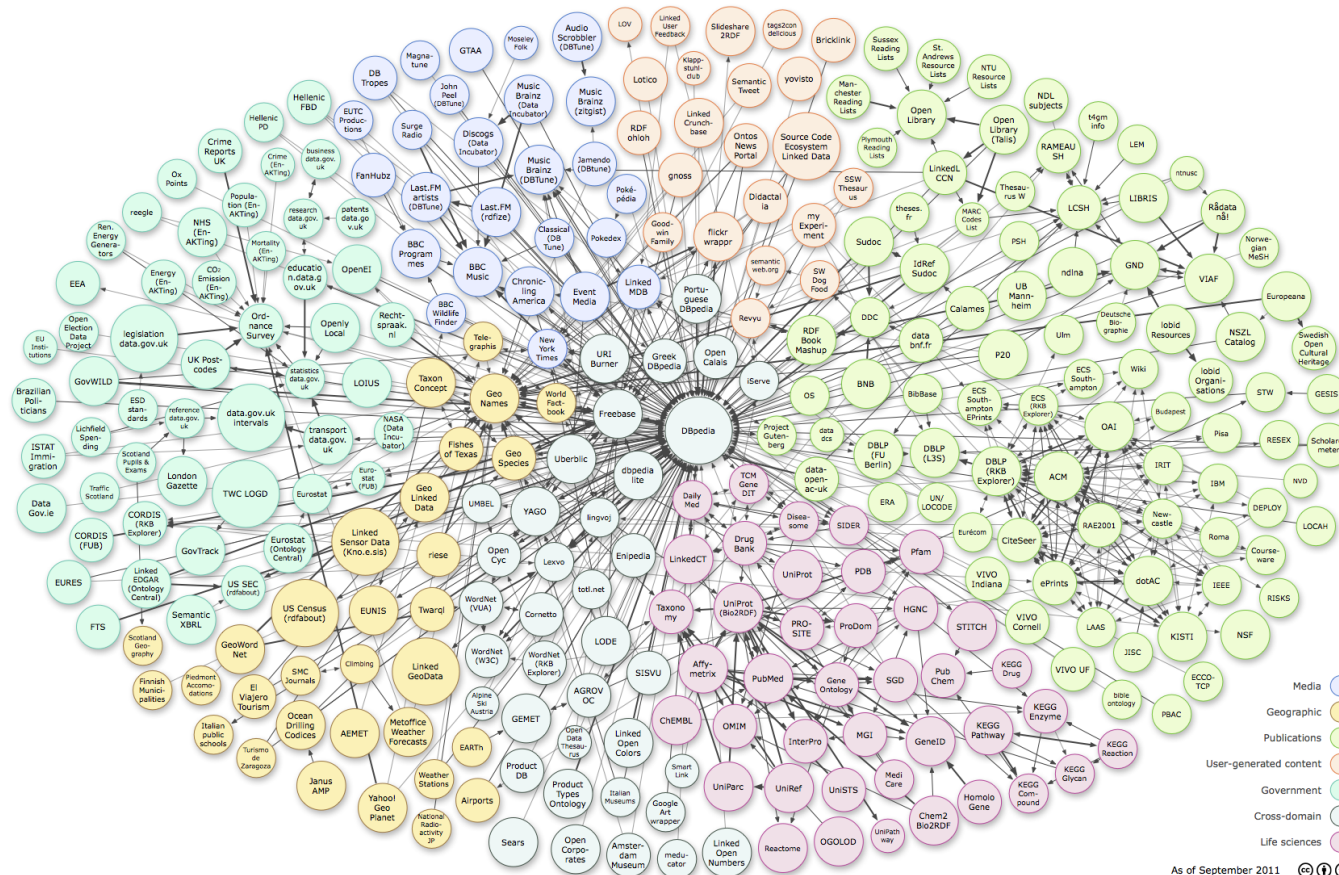
Wer ist Carlo Pedersoli?



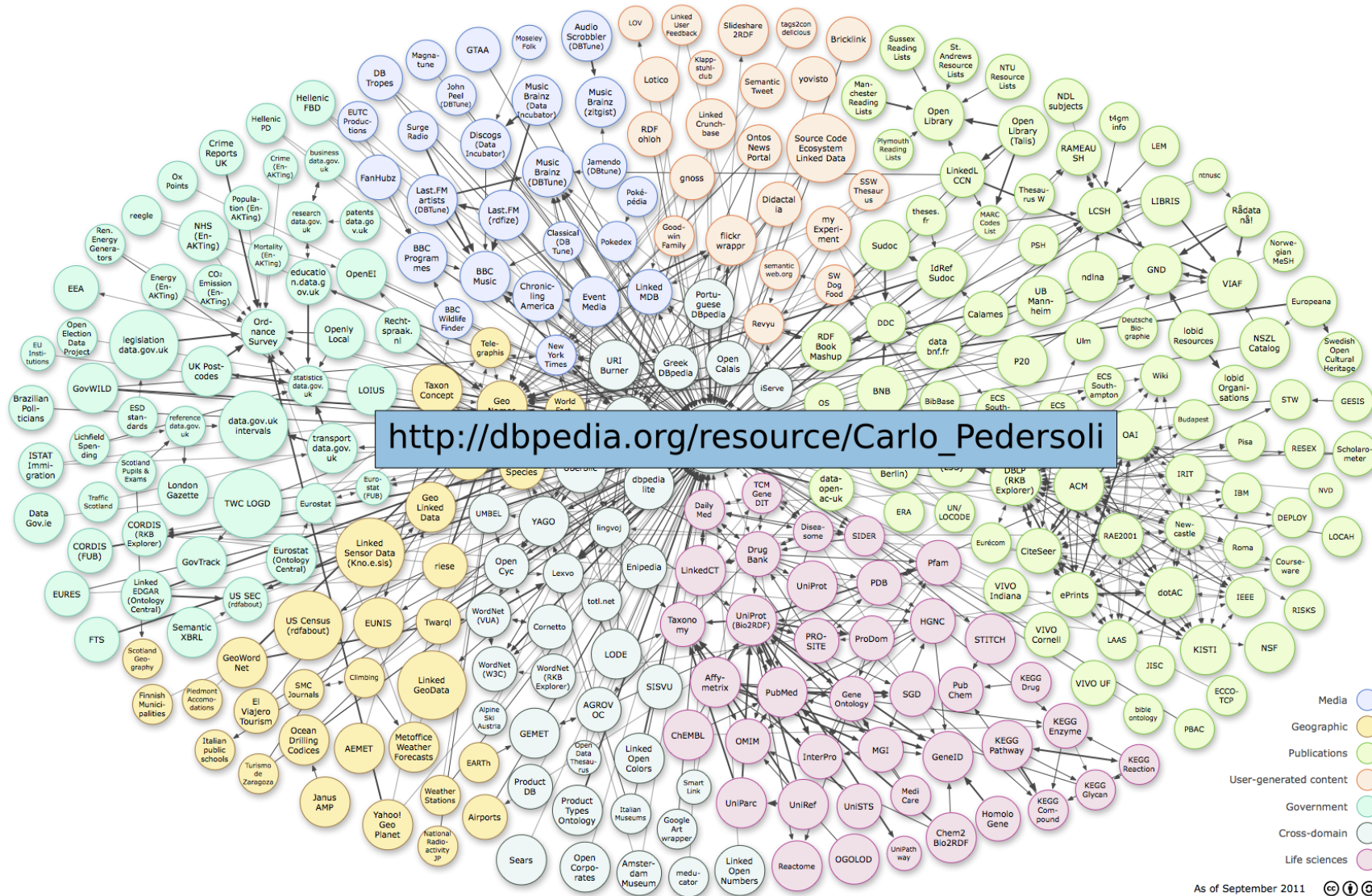
# Wer ist Carlo Pedersoli?

Linked Open Data

Wer ist Carlo Pedersoli?

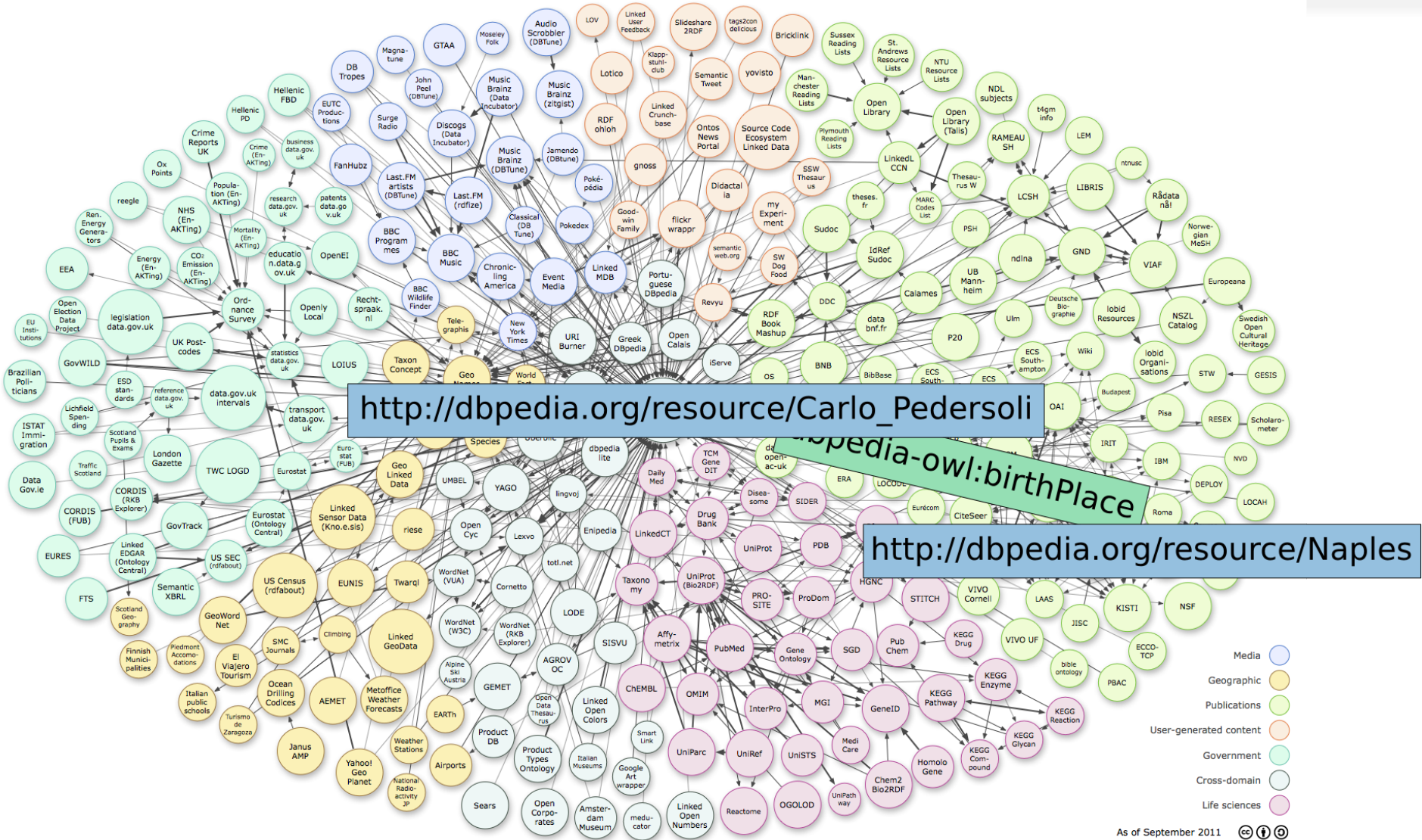


# Wer ist Carlo Pedersoli?

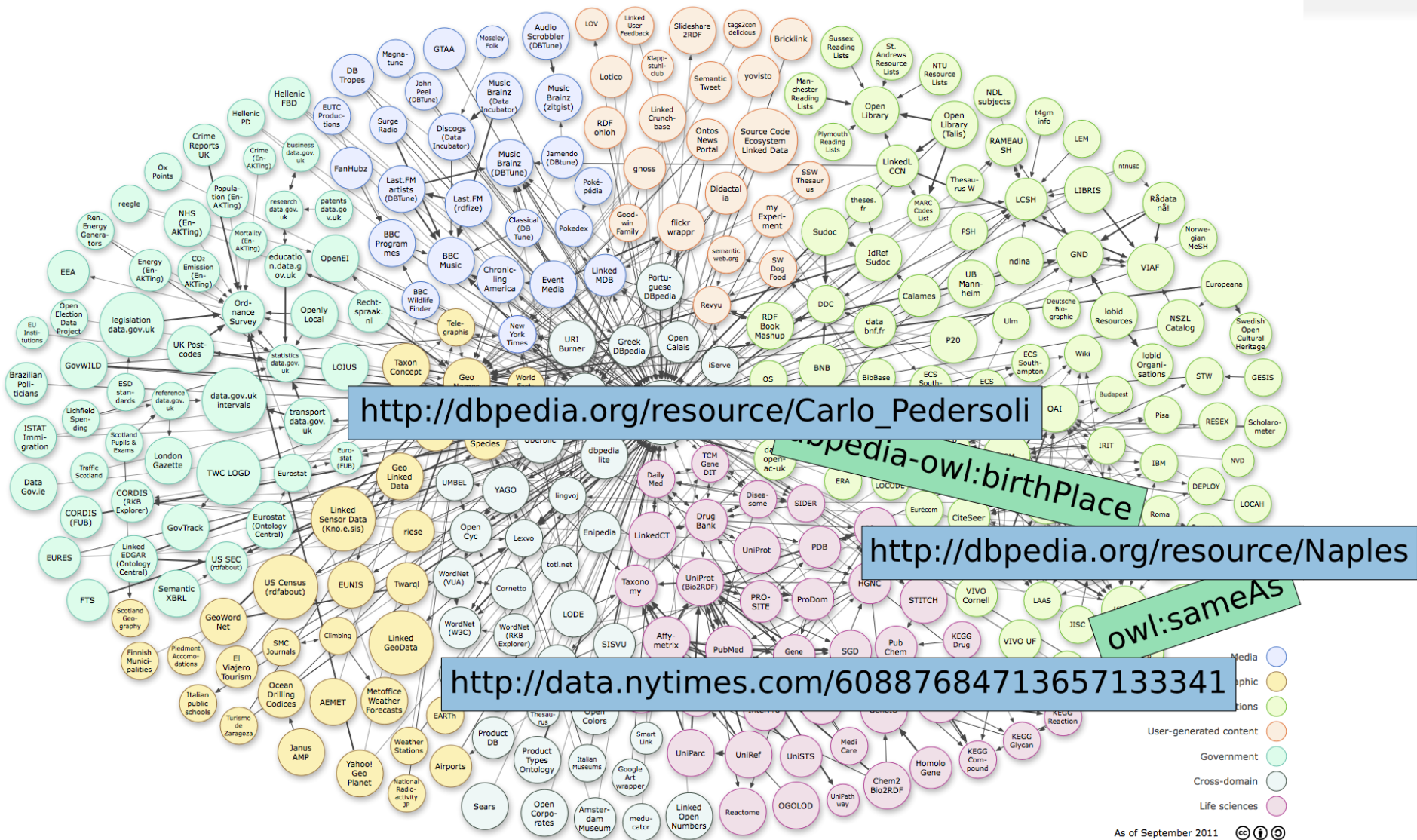




# Wer ist Carlo Pedersoli?

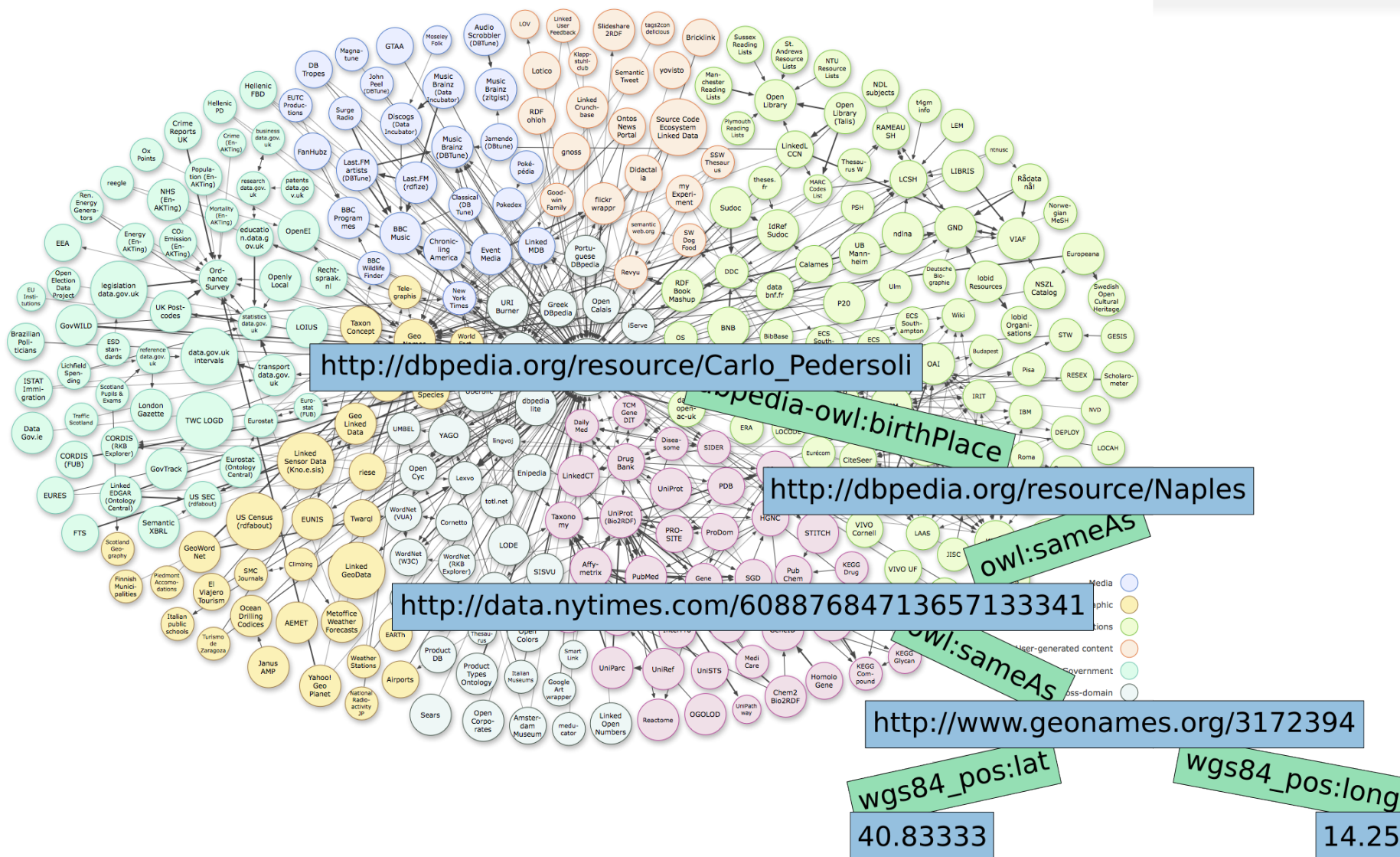


# Wer ist Carlo Pedersoli?

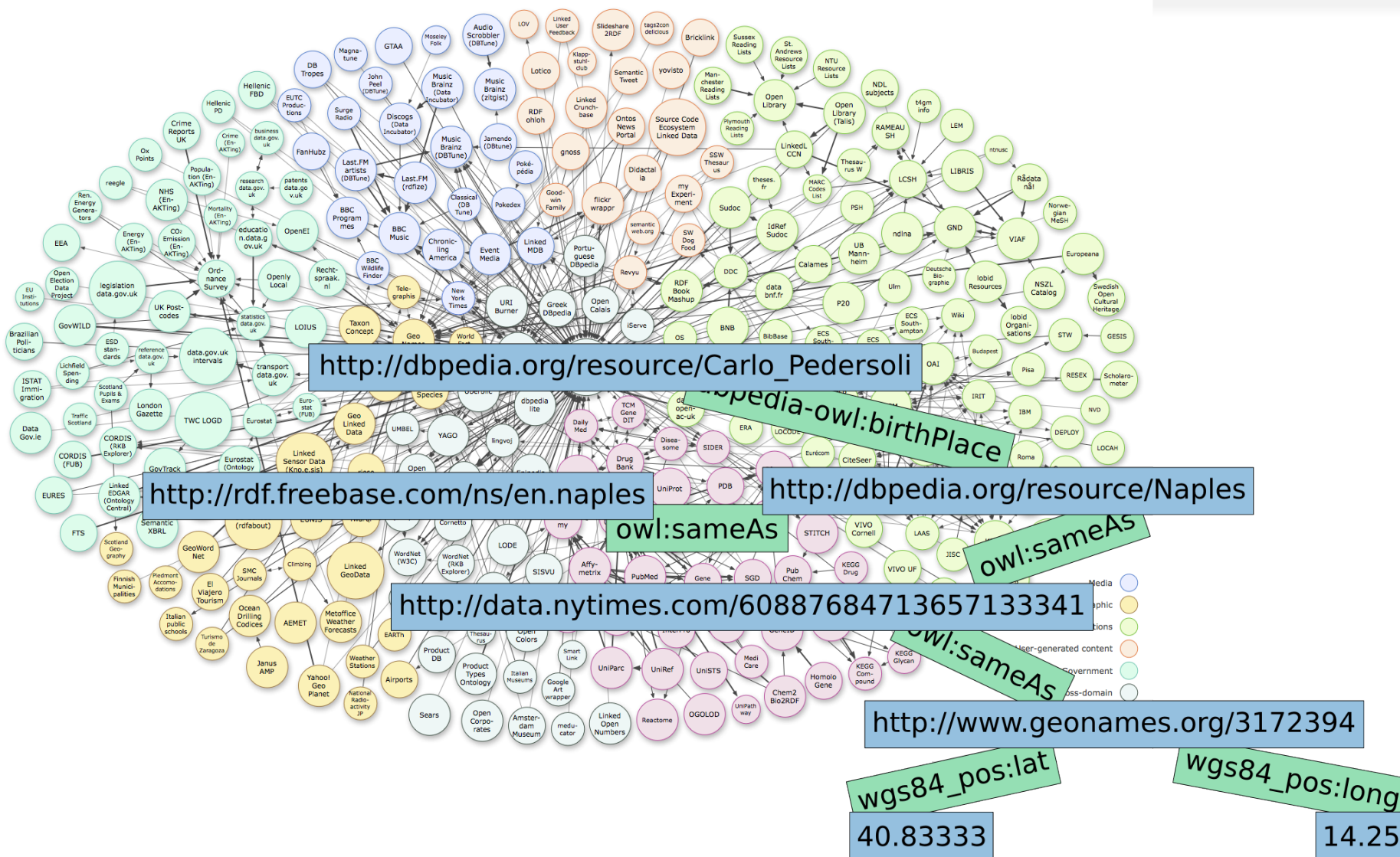




# Wer ist Carlo Pedersoli?

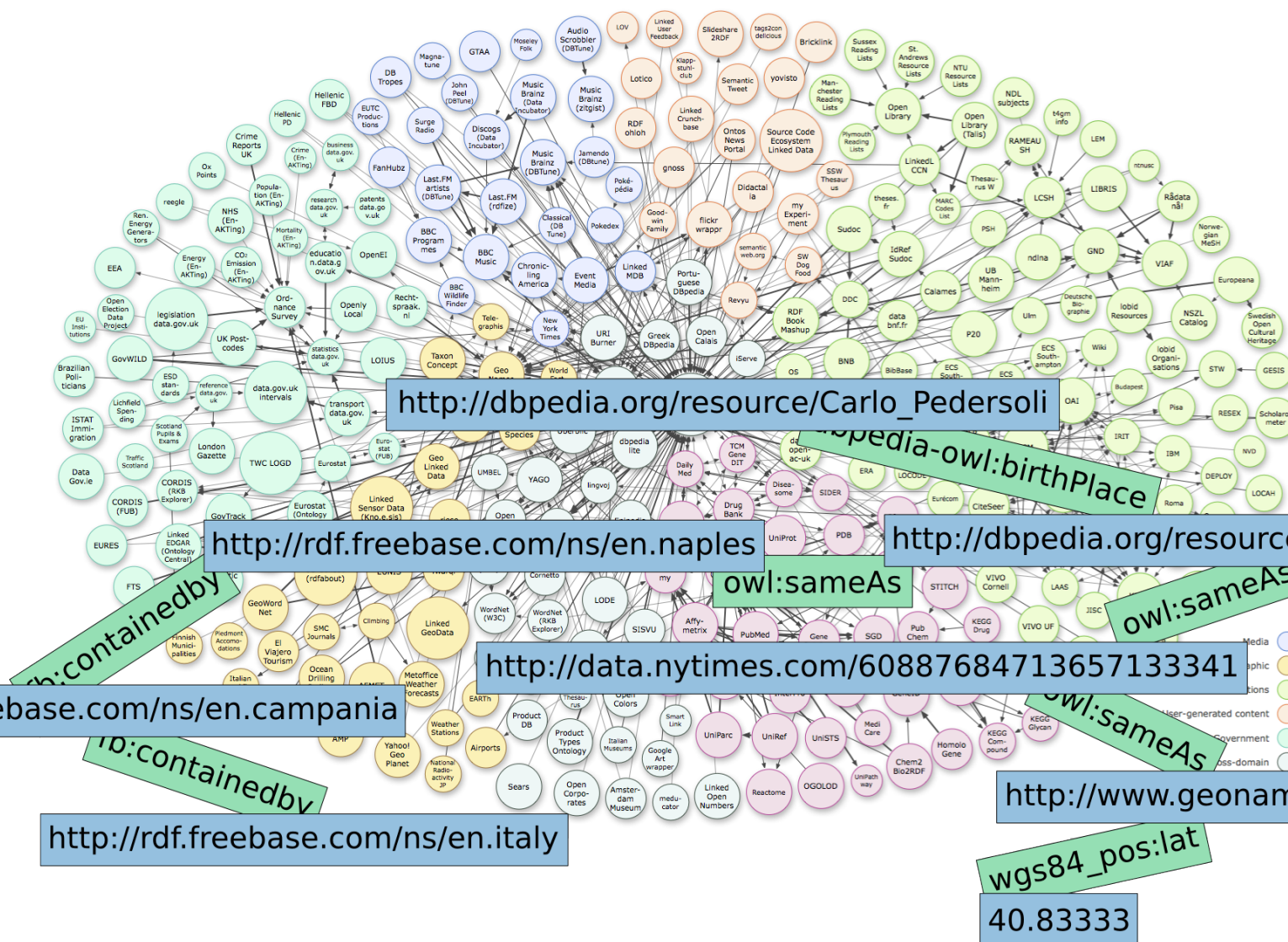


# Wer ist Carlo Pedersoli?

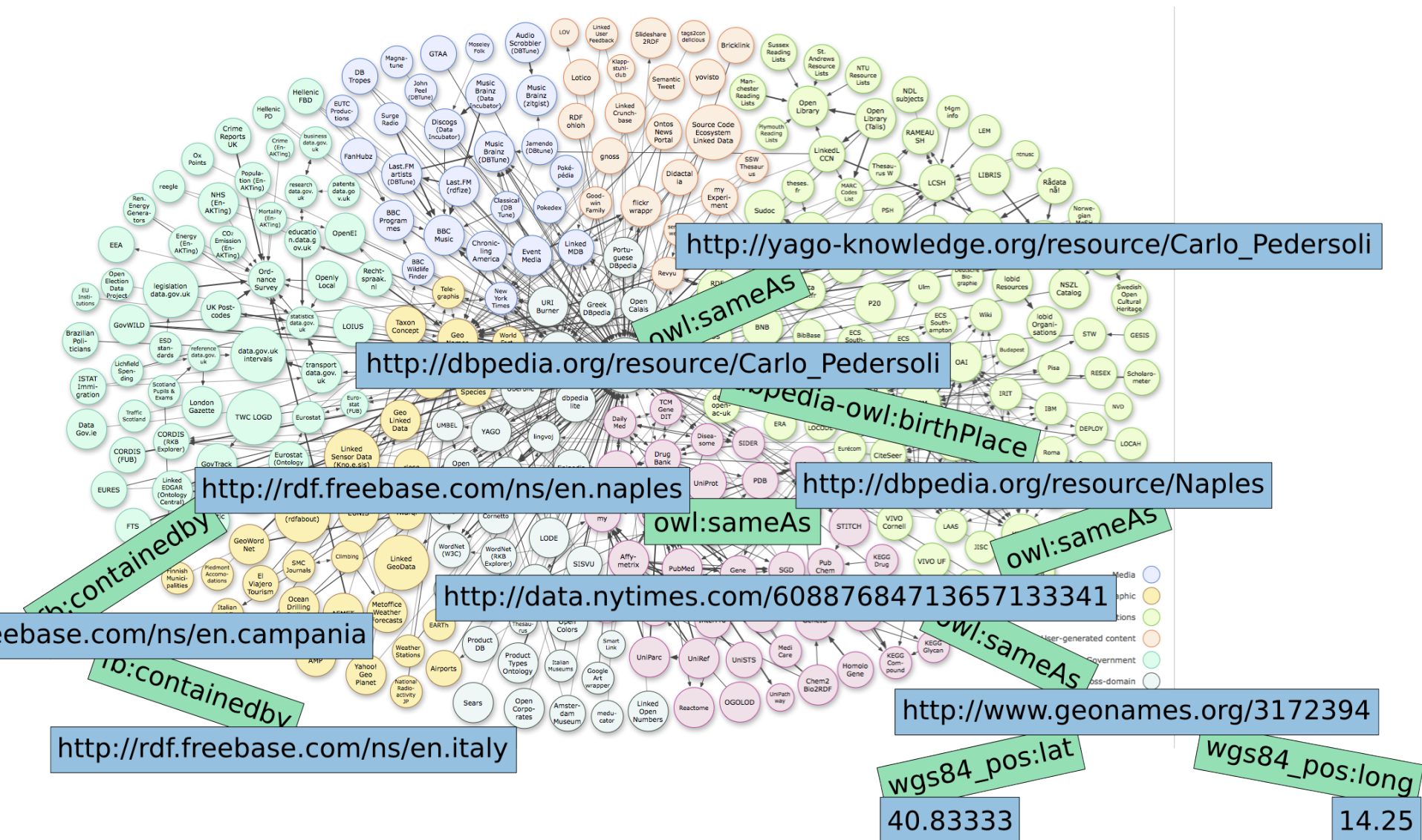




# Wer ist Carlo Pedersoli?

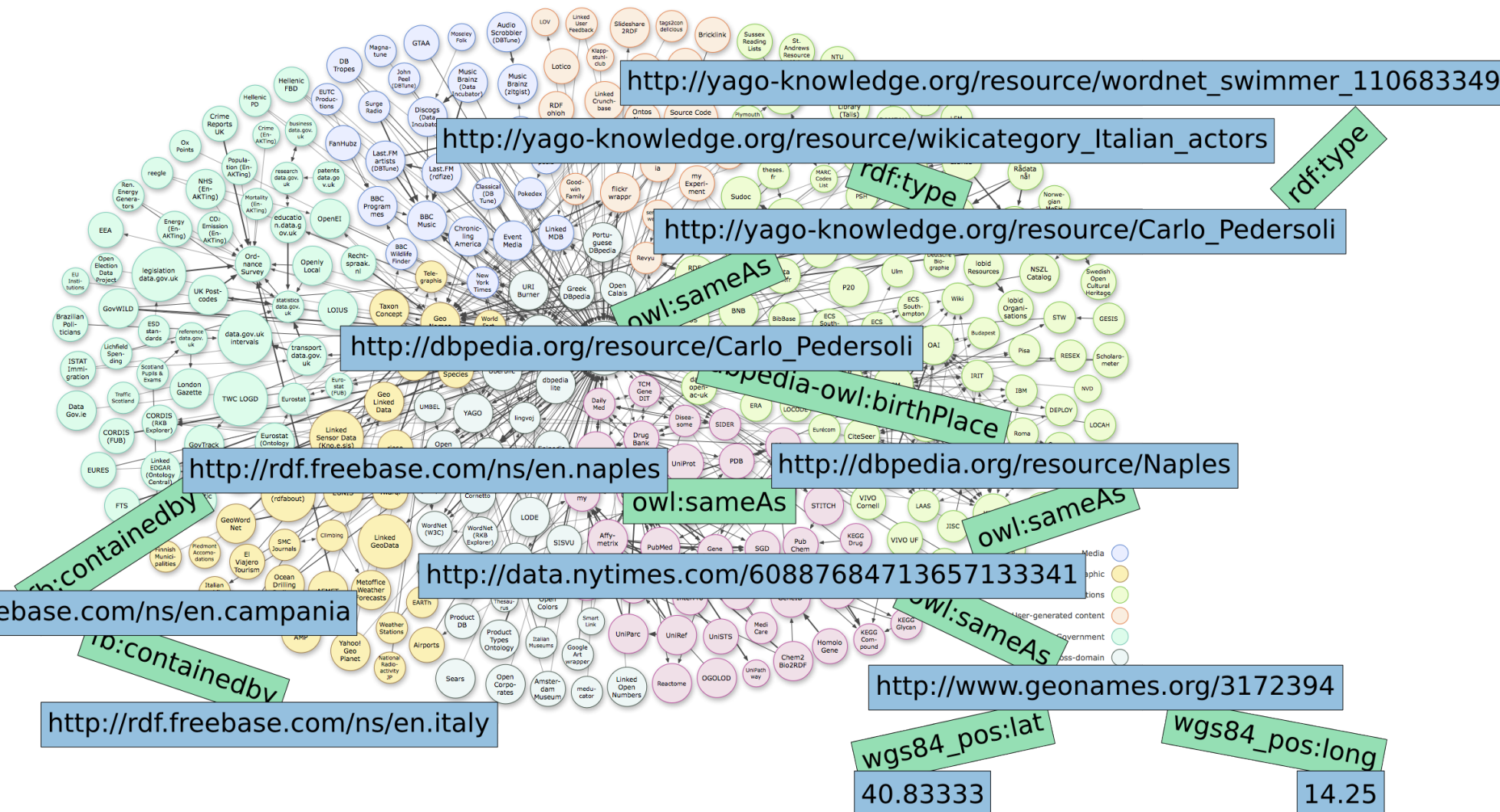


# Wer ist Carlo Pedersoli?

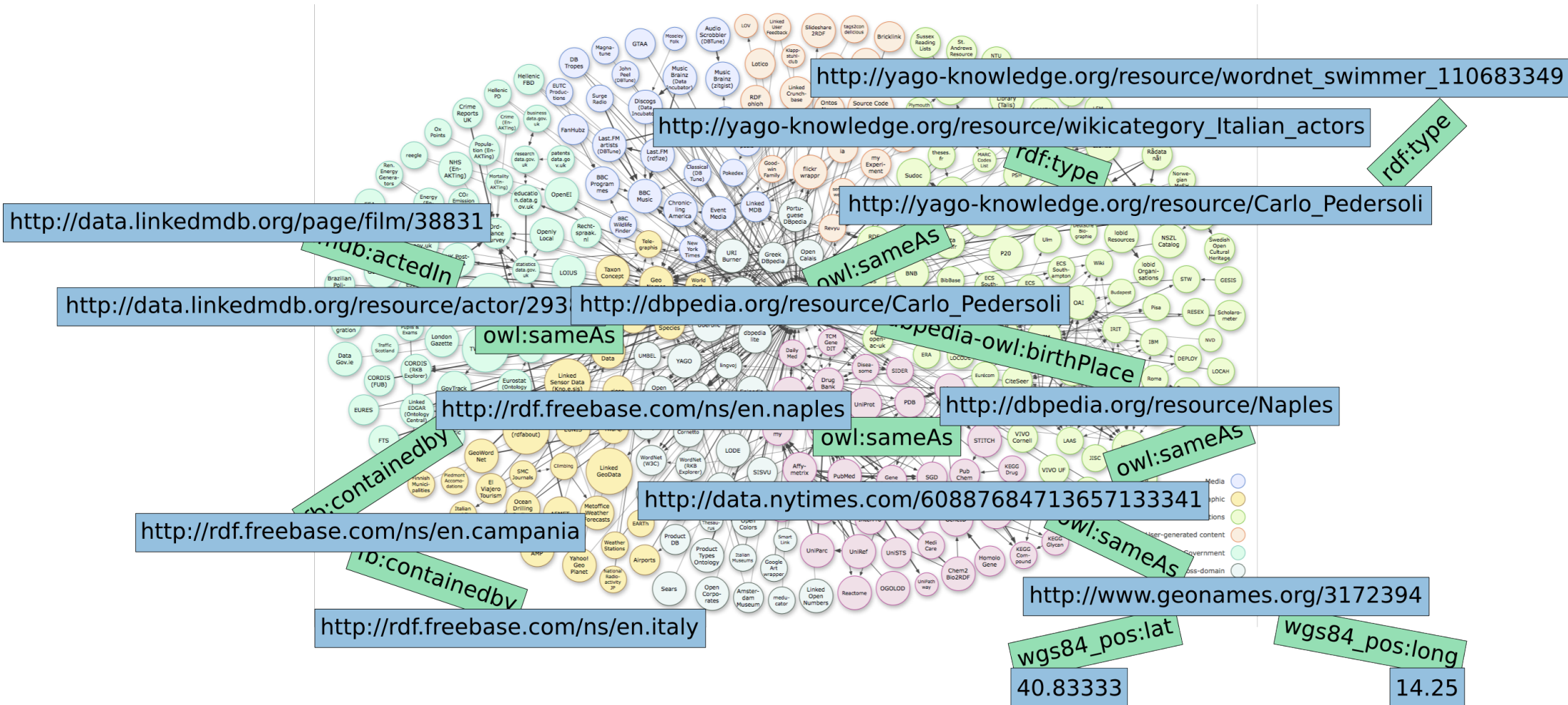




# Wer ist Carlo Pedersoli?



# Wer ist Carlo Pedersoli?



# Wer ist Carlo Pedersoli?

[http://yago-knowledge.org/resource/wordnet\\_actor\\_109765278](http://yago-knowledge.org/resource/wordnet_actor_109765278)

rdf:type

<http://data.linkedmdb.org/resource/actor/29666>

imdb:actedIn

<http://data.linkedmdb.org/page/film/38831>

imdb:actedIn

<http://data.linkedmdb.org/resource/actor/293>

[http://dbpedia.org/resource/Carlo\\_Pedersoli](http://dbpedia.org/resource/Carlo_Pedersoli)

owl:sameAs

<http://rdf.freebase.com/ns/en.naples>

<http://dbpedia.org/resource/Naples>

owl:sameAs

<http://rdf.freebase.com/ns/en.campania>

db:containedby

<http://rdf.freebase.com/ns/en.italy>

db:containedby

<http://data.nytimes.com/60887684713657133341>

owl:sameAs

<http://www.geonames.org/3172394>

wgs84\_pos:lat

40.83333

wgs84\_pos:long

14.25

[http://yago-knowledge.org/resource/wordnet\\_swimmer\\_110683349](http://yago-knowledge.org/resource/wordnet_swimmer_110683349)

[http://yago-knowledge.org/resource/wikicategory\\_Italian\\_actors](http://yago-knowledge.org/resource/wikicategory_Italian_actors)

rdf:type

[http://dbpedia.org/resource/Mario\\_Girotti](http://dbpedia.org/resource/Mario_Girotti)

[http://yago-knowledge.org/resource/Carlo\\_Pedersoli](http://yago-knowledge.org/resource/Carlo_Pedersoli)

owl:sameAs

dbpedia-owl:birthPlace

owl:sameAs

owl:sameAs

owl:sameAs



# Wer ist Carlo Pedersoli?

[http://yago-knowledge.org/resource/wordnet\\_actor\\_109765278](http://yago-knowledge.org/resource/wordnet_actor_109765278)

rdf:type

<http://data.linkedmdb.org/resource/actor/29666>

rdf:subclassOf

[http://yago-knowledge.org/resource/wordnet\\_swimmer\\_110683349](http://yago-knowledge.org/resource/wordnet_swimmer_110683349)

[http://yago-knowledge.org/resource/wikicategory\\_Italian\\_actors](http://yago-knowledge.org/resource/wikicategory_Italian_actors)

rdf:type

[http://dbpedia.org/resource/Mario\\_Girotti](http://dbpedia.org/resource/Mario_Girotti)

rdf:type

<http://data.linkedmdb.org/page/film/38831>

foaf:name

[http://yago-knowledge.org/resource/Carlo\\_Pedersoli](http://yago-knowledge.org/resource/Carlo_Pedersoli)

imdb:actedIn

imdb:hasTitle

imdb:hasAlternativeTitle

They call me Trinity

Die rechte und die linke Hand des Teufels

Lo chiamavano Trinità

Terence Hill

owl:sameAs

[http://dbpedia.org/resource/Carlo\\_Pedersoli](http://dbpedia.org/resource/Carlo_Pedersoli)

Bud Spencer

foaf:name

owl:sameAs

dbpedia-owl:birthPlace

<http://rdf.freebase.com/ns/en.naples>

<http://dbpedia.org/resource/Naples>

db:containedby

owl:sameAs

<http://data.nytimes.com/60887684713657133341>

owl:sameAs

<http://rdf.freebase.com/ns/en.campania>

db:containedby

owl:sameAs

<http://rdf.freebase.com/ns/en.italy>

<http://www.geonames.org/3172394>

wgs84\_pos:lat

40.83333

wgs84\_pos:long

14.25

# Möglichkeiten der Suche

Wie können Benutzer Daten finden?

- Browsing
  - Start-URI
- Suchmaschinen
  - Keywords
- Systeme zur Anfragebearbeitung in heterogenen Datenquellen
  - Strukturierte Anfragen
    - **SPARQL**
    - RDQL
    - SeRQL
    - RQL
    - RDFQL
    - ...

- 1 Motivation
- 2 Suchmaschinen
  - Swoogle
  - Falcons
  - Sindice
  - Sig.ma
  - Indexierung
- 3 Lookup-basierte Anfragebearbeitung
  - Explorative Anfragebearbeitung
  - Index-basierte Anfragebearbeitung
- 4 Systeme zur Anfragebearbeitung in heterogenen Datenquellen
  - Verteilte Datenbanken
  - Naive verteilte Anfragebearbeitung
  - Anfrageoptimierung
- 5 Zusammenfassung

# Suchmaschinen

## Initialisierung

- RDF-Daten aus dem Web crawlen
- Index erstellen

## Benutzeranfragen bearbeiten

- Input: Schlüsselwort
- Im Index suchen (Index Lookup)
- Output: Liste relevanter Quellen

## Probleme

- Updates der Originaldaten

# Suchmaschinen

## Initialisierung

- RDF-Daten aus dem Web crawlen
- Index erstellen

## Benutzeranfragen bearbeiten

- Input: Schlüsselwort
- Im Index suchen (Index Lookup)
- Output: Liste relevanter Quellen

## Probleme

- Updates der Originaldaten



# Suchmaschinen

## Initialisierung

- RDF-Daten aus dem Web crawlen
- Index erstellen

## Benutzeranfragen bearbeiten

- Input: Schlüsselwort
- Im Index suchen (Index Lookup)
- Output: Liste relevanter Quellen

## Probleme

- Updates der Originaldaten

# Typen von Suchmaschinen

## Architekturen

- Lokale Kopie der Daten (Local Copy)
- Ausschließlich Indexe (Index Only)

## Ergebnistypen

- RDF-Dokumente und -Quellen
- RDF-Entitäten und -Konzepte

Manchmal ist die Zuordnung nicht ganz eindeutig möglich, z.B. Caching

# Swoogle

Fokus: Dokumente und Quellen

Crawler, um RDF-Dokumente zu finden

- Beginn zum Beispiel: Google search for „.rdf“
- Metadaten extrahieren  
Encoding (N3, RDF/XML, N-Triple), Language (OWL, RDFS, RDF, DAML),...
- Beziehungen zu anderen Dokumenten bestimmen  
import, extend, references,...
- Google/Jena Crawler
  - Suche nach bestimmten Dokumenttypen (.rdf, .owl, .n3, .daml)
  - Neue Links beim Parsen extrahieren

# Swoogle

## SQL-Datenbank für Metadaten

- Encoding, Language, Statistiken, Beziehung zu anderen Dokumenten (Imports), etc.

## Anfragebearbeitung

- Schlüsselwörter, Field-Constraints (encoding, file type, etc.)
- Ranking ähnlich wie PageRank
- Ergebnis: Links zu Dokumenten

# Swoogle

## SQL-Datenbank für Metadaten

- Encoding, Language, Statistiken, Beziehung zu anderen Dokumenten (Imports), etc.

## Anfragebearbeitung

- Schlüsselwörter, Field-Constraints (encoding, file type, etc.)
- Ranking ähnlich wie PageRank
- Ergebnis: Links zu Dokumenten

# Suchmaske



http://swoogle.umbc.edu/

# Suche nach Naples

The screenshot shows the Swoogle Semantic Web Search Engine interface. The browser address bar displays the URL: [http://swoogle.umbc.edu/index.php?option=com\\_frontpage&service=search&queryType=search\\_swd\\_all&searchString](http://swoogle.umbc.edu/index.php?option=com_frontpage&service=search&queryType=search_swd_all&searchString). The search bar contains the text "Naples". Below the search bar, a blue banner indicates "list documents matching document search" and "1 - 10 of total 1,204 results for Naples in 0.332 seconds". The results are sorted by date and triple. The first result is a SemanticWebDocument from Stanford University. The second result is a SemanticWebDocument (embedded) from MyWorld-Travel.com. The third result is a SemanticWebDocument (embedded) from Flickr. The fourth result is a SemanticWebDocument (embedded) from MyWorld-Travel.com. The fifth result is a SemanticWebDocument (embedded) from MyWorld-Travel.com. The sixth result is a SemanticWebDocument from MyWorld-Travel.com. The seventh result is a SemanticWebDocument from the New York Times.

Want more results? [Login](#)

**Swoogle** [ontology](#) [document](#) [term](#) [across ontologies](#)

semantic web search  [Swoogle Search](#) [RDF version](#)

list documents matching document search 1 - 10 of total 1,204 results for Naples in 0.332 seconds

sort by [date](#) | [triple](#) |

<http://tap.stanford.edu/data/>  
SemanticWebDocument, RDFXML, 2006-03-24, 15M, ontoRatio(0.14), [metadata](#), [cached](#)

<http://www.myworld-travel.com/wordpress/world-travel/naples-to-amalfi-coast-transfers-tours>  
[DESC] title - **Naples** to Amalfi coast: transfers & tours  
SemanticWebDocument (embedded), RDFXML, 2009-03-02, 57K, [metadata](#), [cached](#)

<http://www.flickr.com/photos/bright/130232105/>  
[DESC] description - **Naples**, April 2006 | title - birds on a wire | date - 2006  
SemanticWebDocument (embedded), RDFXML, 2009-03-09, 127K, [metadata](#), [cached](#)

<http://www.myworld-travel.com/wordpress/world-travel/naples-to-amalfi-coast-transfers-tours/>  
SemanticWebDocument (embedded), RDFXML, 2009-08-09, 62K, [metadata](#), [cached](#)

<http://www.myworld-travel.com/wordpress/2391/naples-to-amalfi-coast-transfers-tours/>  
[DESC] title - **Naples** to Amalfi coast: transfers & tours  
SemanticWebDocument (embedded), RDFXML, 2009-08-10, 62K, [metadata](#), [cached](#)

<http://www.myworld-travel.com/wordpress/2391/naples-to-amalfi-coast-transfers-tours/feed/atom/>  
SemanticWebDocument, RDFXML, 2009-08-21, 877 B, [metadata](#), [cached](#)

<http://data.nytimes.com/9981562425865839491>  
[DESC] number\_of\_variants - 1 | prefLabel - **Naples** (Fla) | long - -81.7948103 | associated\_article\_count  
SemanticWebDocument, RDFXML, 2010-06-22, 8K, [metadata](#), [cached](#)

Fertig

<http://swoogle.umbc.edu/>

# Falcons

Fokus: Entitäten

Crawling

- Folge den in RDF-Dokumenten enthaltenen Links
- Erstelle virtuelle Dokumente für Entitäten
  - Literals
  - Human-readable Names and Descriptions (rdfs:label, rdfs:comment)
- Indexe erstellen
  - Terme in virtuellen Dokumenten
  - Klassen von Entitäten
- RDF-Triples in einer Datenbank hinterlegen



# Falcons

## Anfragebearbeitung

- Keywords
- Filterung durch Klassen/Typen
- Ranking basierend auf Ähnlichkeit zur Keyword-Anfragen und Popularität
- Output: Entitäten mit Textausschnitten (Snippets)

# Suchmaske

The screenshot shows a web browser window titled "Falcons Object Search" with the URL `http://ws.nju.edu.cn/falcons/objectsearch/index.jsp`. The browser's address bar and search bar are visible. Below the browser window, the main content area displays the "Falcons" logo, which features a magnifying glass over the letter 'o'. Underneath the logo, there are links for "Object", "Concept", "Ontology", and "Document". A search input field is present, followed by a "Search Objects" button. At the bottom of the page, there are links for "Submit URI", "API", "FAQ", and "Contact us", along with a copyright notice: "©2011 Websoft Research Group, Nanjing University, P.R. China".

Falcons

Object [Concept](#) [Ontology](#) [Document](#)

Search Objects

[Submit URI](#) - [API](#) - [FAQ](#) - [Contact us](#)

©2011 [Websoft Research Group](#), Nanjing University, P.R. China

Fertig

`http://ws.nju.edu.cn/falcons/objectsearch/index.jsp`

## Suchmaske

The screenshot shows a web browser window titled "Bud Spencer - Falcons Object Search" with the URL <http://ws.nju.edu.cn/falcons/objectsearch/result.jsp?query=Bud+Spencer>. The page features the "Falcons" logo and navigation links for "Object", "Concept", "Ontology", and "Document". A search bar contains the text "Bud Spencer" and a "Search Objects" button.

On the left, a sidebar titled "Type" lists various categories: Any type, agent, artifact, concept, creation, document, entity10001740, event, film, performance, person, physical entity, physical object, product, psychological feature100023100, show, social entity, social event, spatial thing, unit, and work.

The main content area displays "Objects 1 - 5 of 100 for your search Bud Spencer". The first result is "Bud Spencer - Actor, Person, actor, LivingPeople, person, actor, Person, Thing, Artist, player", which includes properties like type, label, sameAs, name, comment, hasPhotoCollection, subject, reference, wiki page uses template, and abstract. The second result is "Bud Spencer Blues Explosion - music artist", and the third is "Photos for DBpedia.org resource Bud Spencer - Document".

The status bar at the bottom left indicates "Fertig".

<http://ws.nju.edu.cn/falcons/objectsearch/index.jsp>

## Suchmaske

The screenshot shows a web browser window titled "Bud Spencer - Falcons Object Search". The address bar shows the URL: [http://ws.nju.edu.cn/falcons/objectsearch/result.jsp?query=Bud+Spencer&type\\_uri\\_id=23,61177992,31061847,1161](http://ws.nju.edu.cn/falcons/objectsearch/result.jsp?query=Bud+Spencer&type_uri_id=23,61177992,31061847,1161). The page features the "Falcons" logo and navigation links: Object, Concept, Ontology, Document. A search bar contains "Bud Spencer" and a "Search Objects" button.

On the left, a sidebar titled "Type" lists various categories: Any type, person, actor, athlete, athlete at the 1928 summer olympics, athlete of the united states, baseball player, baseball team, contestant, entertainer, human, jock, major league, major league baseball team, performer, person, player, upper middle class, worker.

The main content area displays "Objects 1 - 5 of 100 for your search Bud Spencer". The first result is "Bud Spencer - Actor, Person, actor, LivingPeople, person, actor, Person, Thing, Artist, player". It includes properties like type, label, sameAs, name, comment, hasPhotoCollection, subject, reference, and wiki page uses template. The second result is "Bud Spencer (Actor) - Person, actor", which includes properties like type, label, is primary topic of, page, performance, is actor of, actor\_name, actor\_actorid, actor\_netflix\_id, and actor\_nytimes\_id. The third result is "Emerson Spencer - athlete of the United States, athlete at the 1928 Summer Olympics, OlympicGoldMedalistsForTheUnitedStates", which includes comment and abstract properties.

The status bar at the bottom indicates "Fertig".

<http://ws.nju.edu.cn/falcons/objectsearch/index.jsp>

# Sindice

Fokus: Dokumente und Quellen

Indexe

- URI (Unified Resource Identifier)
- IFP (Inverse Functional Properties)
- Literal

Features

- Reasoning
- SPARQL endpoint indexing

# Suchmaske

The screenshot shows the Sindice website, titled "Sindice - The semantic web index". The browser address bar shows "http://sindice.com/". The website has a navigation bar with links: About, Search, Submit Your Data, Jobs, Support, and Dev. The main content area is divided into several sections:

- Sindice - Data Web Services:** A blue banner with the text "Billion pieces of reusable information can already be found across hundreds of millions web pages which embed RDF and Microformats. Start consuming this data today with Sindice Data Web services." and a "LEARN MORE" button.
- Search:** A green box with tabs for "Search", "Sparql", "Submit", and "Inspector Tool". It includes a search input field, a "SEARCH" button, and examples: "tim berners lee (by URI), michele, deri". It also states "Searching on about 474.68 million documents."
- LATEST DATA:** A list of recent data entries with timestamps and triple counts, such as "10:56:51 7 triples http://yenile.org...".
- SINDICE IS HIRING:** A section with the text "Excited by the Web of Data? Sindice is hiring!" and a "Apply with LinkedIn" button.
- HIGHLIGHTS:** A section featuring "SIREn { Semantic Information Retrieval Engine }" with links to Home, Features, News, Download, Documentation, and Contact.
- SINDICE BLOG:** A section with a "Feed" icon and several blog posts, including "Sig.ma Enterprise Edition (EE) available" dated AUG 24, 2011, and "Sindice, its startup company and 12 billion+ live triples SPARQL endpoint" dated JUN 14, 2011.

<http://sindice.com>

# Suchmaske

The screenshot shows a web browser window titled "Search the Data Web - Sindice" with the URL <http://sindice.com/search?q=Bud+Spencer>. The page features the Sindice logo and navigation links: About, Search, Submit Your Data, Jobs, Support, and Dev. The search interface includes a search bar with the keyword "Bud Spencer", a "SEARCH" button, and options for "Group By Dataset" and "Sorted by: relevance".

On the left side, there are filters for "Quick filters (All options)", "Time range" (Any date, Today, Yesterday, Last week, Last month, Last year), "Format" (Any format, RDF, RDFa, Microdata, Microformat, XFN, HCard, HCalendar, HListing, HResume, License, Geo, ADR), "Predicate" (e.g., label OR dc:title), "Class" (e.g., foaf:person AND student), "Ontology" (e.g., foaf), and "Domain" (e.g., abc.com OR abc).

The search results show 4,099 documents found in 0.03 seconds. The results are listed as follows:

- "Bud Spencer"** (RDFa)
  - 2011-07-16 – 7 triples in 787 Bytes
  - <http://www.webnews.de/257867/bud-spencer> (Search) Inspect: (Cache) (Live)
- "Bud Spencer"** (TITLE, XFN, RDFa, LICENSE)
  - 2012-01-09 – 7 triples in 758 Bytes
  - <http://www.machacas.org/tag/bud-spencer> (Search) Inspect: (Cache) (Live)
- Bud spencer blues explosion** (RDFa)
  - 2010-07-16 – 35 triples in 7.5 kB
  - <http://it.bestshopping.com/prezzi/Bud-spencer-blues-explosion.sku=8016670273240%7C.html> (Search) Inspect: (Cache) (Live)
- Bud spencer terence hill** (RDFa)
  - 2010-07-14 – 35 triples in 7.3 kB
  - <http://it.bestshopping.com/prezzi/Bud-spencer-terence-hill.sku=9788884401380%7C.html> (Search) Inspect: (Cache) (Live)
- "Pictures of Bud Spencer"** (RDFa)
  - 2011-07-12 – 27 triples in 3.9 kB
  - [http://lookaboo.com/o/pictures/topic/2424495/Bud\\_Spencer](http://lookaboo.com/o/pictures/topic/2424495/Bud_Spencer) (Search) Inspect: (Cache) (Live)
- Bud Spencer - Cinebel** (HCard, RDFa)
  - 2011-09-12 – 14 triples in 1.4 kB
  - [http://www.cinebel.be/nl/person/756/Bud\\_Spencer](http://www.cinebel.be/nl/person/756/Bud_Spencer) (Search) Inspect: (Cache) (Live)
- Bud Spencer - Cinebel** (HCard, RDFa)

The status bar at the bottom of the browser window shows "Fertig".



# Sig.ma

Fokus: Browser und Mashup-Generator

Funktionsweise

- Quellen finden
- Daten extrahieren
- Gruppieren und ausgeben

# Sig.ma

Sig.ma EE- Semantic Information Mashup Enterprise Edition

http://sig.ma/

Meistbesuchte Seit... Erste Schritte Aktuelle Nachricht...

Sig.ma EE- Semantic Information ...

**SIG.MA**  
SEMANTIC INFORMATION  
MASHUP

Help About Support

Version: 2.0.8

**Sig.ma - Live views on Web(s) of Data**

Sig.ma does on the fly, interactive information visualization with bits coming from up to hundreds of sources at the same time. Sig.ma pages have permalinks and can be embedded in web pages.

Use it online or [Download Sig.ma EE](#)

**Search on our live Sig.ma installation:**

Type one or more keywords [Search](#)

Use: ☒ Sindice ☒ OKKAM  
☒ YBoss ☒ Lod Sparql Endpoint ☐ Your own data

Examples: [Tim Berners Lee](#), [Barack Obama](#), [Michael Jackson](#)

**Sig.ma - Live views on the Web of Data**

from Sindice Team

**SIG.MA**  
SEMANTIC INFORMATION  
MASHUP

- Browser
- Mashup Generator
- API

... For the Web of Data

07:11 [vimeo](#)

Übertragen der Daten von b.scorecardresearch.com...

**Sig.ma is powered by:**

**indice**  
THE SEMANTIC WEB INDEX

**LOD2** **OKKAM** **BOSS**

More in the [about](#). Much more in [our blog](#).

While Sig.ma is by no mean the first data aggregator for the Semantic Web, its contribution is to show that the sum is really bigger than the single parts and exciting possibilities lie in a holistic approach for automatic semistructured data discovery and consolidation. [Read more](#)

<http://sig.ma>

## Sig.ma

Sig.ma EE- Semantic Information Mashup Enterprise Edition

http://sig.ma/search?q=Bud+Spencer

Meistbesuchte Seit... Erste Schritte Aktuelle Nachricht...

Sig.ma EE- Semantic Information ...

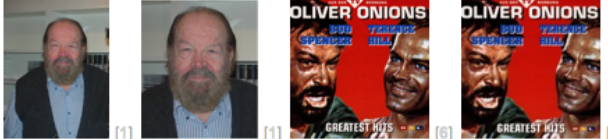
**SIG.MA** SEMANTIC INFORMATION MASHUP

Help About Support

Version: 2.0.8

Bud Spencer Add More Info Start New Order Options Use It

### Bud Spencer & Terence Hill Greatest Hits, Volume 1

picture: 

given name: Bud [1]

family name: Spencer [1]

comment: Actor [1]

Bud Spencer (born Carlo Pedersoli on 31 October 1929) is an Italian actor, filmmaker and former swimmer (he was the first Italian to swim 100m in less than a minute). He is known for past roles in spaghetti westerns together with his long time filmpartner Terence Hill. Growing from a successful swimmer in his youth, he got a degree in law, and has registered several patents. [1]

Bud Spencer (born Carlo Pedersoli on 31 October 1929) is an Italian actor, filmmaker, former swimmer (he was the first Italian to swim 100m in less than a minute). He is known for his weight (he was about 130 kg in 70s and exceeded 150 kg in 90s: today, he is about 120 kg), and for past roles in spaghetti westerns. Growing from a successful swimmer in his youth, he got a degree in law, and has registered several patents. [2]

is albums of: <http://freebase.com/quid/9202a8c04000641f80000000012091ce> [13,14,16,20]

artist: <http://freebase.com/quid/9202a8c04000641f800000000356468c> [12,13,14,15,16,17,20]

album: <http://freebase.com/quid/9202a8c04000641f80000000035645c7> [12]  
<http://freebase.com/quid/9202a8c04000641f800000000356460f> [15]  
<http://freebase.com/quid/9202a8c04000641f8000000006e19ec0> [17]

article: <http://freebase.com/quid/9202a8c04000641f80000000000a1dc> [7]

is actor of: show 11 values

albummeta <http://ec2.images-amazon.com/images/P/B000025FR4.03.MZZZZZZZ.jpg> [6]

coverarturl:

Fertig

Sources (20) Approved (0) Rejected (0)

- Bud Spencer** 104 facts | 2011-10-21  
[http://dbpedia.org/resource/Bud\\_Spencer](http://dbpedia.org/resource/Bud_Spencer)
- Bud Spencer** 11 facts | 2012-01-14  
<http://uberblic.org/resource/483ad20b-1b0...>
- Bud Spencer** 8 facts | 2012-01-14  
<http://uberblic.org/resource/e7c72dab-146...>
- Oliver Onions Bud Spence...** 13 facts | 2012-01-14  
<http://freebase.com/quid/9202a8c04000641f...>
- Oliver Onions Bud Spence...** 13 facts | 2012-01-14  
<http://freebase.com/quid/9202a8c04000641f...>
- Bud Spencer/Terence Hil...** 21 facts | 2012-01-14  
<http://dbtune.org/musicbrainz/resource/re...>
- Bud Spencer** 18 facts | 2012-01-14  
<http://freebase.com/quid/9202a8c04000641f...>
- Bud Spencer** 5 facts | 2012-01-14  
<http://uberblic.org/resource/3dbe24be-edc...>
- Bud Spencer** 8 facts | 2012-01-14  
<http://uberblic.org/resource/d7f737f4-507...>
- Bud Spencer** 4 facts | 2012-01-14  
<http://uberblic.org/resource/fd84ae0c-14f...>
- Bud Spencer** 6 facts | 2012-01-14  
<http://lastfm.rdfize.com/artists/Bud+Spen...>
- Bud Spencer & Terence Hi...** 11 facts | 2012-01-14  
<http://freebase.com/quid/9202a8c04000641f...>

select all approve all

http://sig.ma

# Indexe

## Verschiedene Arten

- „Vollständige“ Indexierung
  - Triple liegen vollständig vor
  - Effizienter Zugriff auf lokale Daten
- Zusammenfassende Indexe
  - Auswahl von relevanten Quellen
  - Kompromiss: Detaillierungsgrad vs. Speicherplatz

# Zusammenfassende Indexe

## Indexe auf Schemaebene

- Properties (URIs die als Prädikate benutzt werden)
- Classes (Objekte in `rdf:type` Triples)

## Beispiel

- 1 `http://data.linkedmdb.org/resource/actor/29388 owl:sameAs http://rdf.freebase.com/ns/m/018gm`
- 2 `http://data.linkedmdb.org/resource/actor/29388 rdf:type http://data.linkedmdb.org/resource/movie/actor`
- 3 `http://rdf.freebase.com/ns/m/018gm fb:place_of_birth http://rdf.freebase.com/ns/en/naples`

| Search Key  | Sources  |
|---|----------|
| <code>owl:sameAs</code>                                     | imdb     |
| <code>rdf:type</code>                                       | imdb     |
| <code>http://data.linkedmdb.org/resource/movie/actor</code> | imdb     |
| <code>fb:place_of_birth</code>                              | freebase |

# Zusammenfassende Indexe

## Indexe auf Schemaebene

- Properties (URIs die als Prädikate benutzt werden)
- Classes (Objekte in `rdf:type` Triples)

## Beispiel

- 1 `http://data.linkedmdb.org/resource/actor/29388 owl:sameAs http://rdf.freebase.com/ns/m/018gm`
- 2 `http://data.linkedmdb.org/resource/actor/29388 rdf:type http://data.linkedmdb.org/resource/movie/actor`
- 3 `http://rdf.freebase.com/ns/m/018gm fb:place_of_birth http://rdf.freebase.com/ns/en/naples`

| Search Key  | Sources  |
|---|----------|
| <code>owl:sameAs</code>                                     | imdb     |
| <code>rdf:type</code>                                       | imdb     |
| <code>http://data.linkedmdb.org/resource/movie/actor</code> | imdb     |
| <code>fb:place_of_birth</code>                              | freebase |

# Zusammenfassende Indexe

## Invertierte URI Indexe

- Alle URIs, die in einer Quelle vorkommen

### Beispiel

- 1 `http://data.linkedmdb.org/resource/actor/29388 owl:sameAs  
http://rdf.freebase.com/ns/m/018gm`
- 2 `http://data.linkedmdb.org/resource/actor/29388 rdf:type  
http://data.linkedmdb.org/resource/movie/actor`
- 3 `http://rdf.freebase.com/ns/m/018gm fb:place_of_birth  
http://rdf.freebase.com/ns/en/naples`

| Search Key  | Sources        |
|---|----------------|
| <code>http://data.linkedmdb.org/resource/actor/29388</code> | imdb           |
| <code>owl:sameAs</code>                                     | imdb           |
| <code>http://rdf.freebase.com/ns/m/018gm</code>             | imdb, freebase |
| <code>rdf:type</code>                                       | imdb           |
| <code>http://data.linkedmdb.org/resource/movie/actor</code> | imdb           |
| <code>fb:place_of_birth</code>                              | freebase       |
| <code>http://rdf.freebase.com/ns/en/naples</code>           | freebase       |

# Zusammenfassende Indexe

## Invertierte URI Indexe

- Alle URIs, die in einer Quelle vorkommen

## Beispiel

- 1 `http://data.linkedmdb.org/resource/actor/29388 owl:sameAs  
http://rdf.freebase.com/ns/m/018gm`
- 2 `http://data.linkedmdb.org/resource/actor/29388 rdf:type  
http://data.linkedmdb.org/resource/movie/actor`
- 3 `http://rdf.freebase.com/ns/m/018gm fb:place_of_birth  
http://rdf.freebase.com/ns/en/naples`

| Search Key  | Sources        |
|---|----------------|
| <code>http://data.linkedmdb.org/resource/actor/29388</code> | imdb           |
| <code>owl:sameAs</code>                                     | imdb           |
| <code>http://rdf.freebase.com/ns/m/018gm</code>             | imdb, freebase |
| <code>rdf:type</code>                                       | imdb           |
| <code>http://data.linkedmdb.org/resource/movie/actor</code> | imdb           |
| <code>fb:place_of_birth</code>                              | freebase       |
| <code>http://rdf.freebase.com/ns/en/naples</code>           | freebase       |



# Suchmaschinen sind nicht genug...

## Anfrage

- In welchen Filmen haben Carlo Pedersoli (Bud Spencer) und Mario Girotti (Terence Hill) zusammen gespielt?
- Oder in welchen Filmen hat nur einer von beiden (ohne den jeweils anderen) gespielt?

# Suchmaschinen sind nicht genug...

## Anfrage

- In welchen Filmen haben Carlo Pedersoli (Bud Spencer) und Mario Girotti (Terence Hill) zusammen gespielt?
- Oder in welchen Filmen hat nur einer von beiden (ohne den jeweils anderen) gespielt?

## Anfragemöglichkeiten

- „Carlo Pedersoli Mario Girotti“
- „Bud Spencer Terence Hill“
- „Bud Spencer Mario Girotti“
- <http://data.linkedmdb.org/resource/actor/29388>  
<http://data.linkedmdb.org/resource/actor/29666>
- ...

# Suchmaschinen sind nicht genug...

## Anfrage

- In welchen Filmen haben Carlo Pedersoli (Bud Spencer) und Mario Girotti (Terence Hill) zusammen gespielt?
- Oder in welchen Filmen hat nur einer von beiden (ohne den jeweils anderen) gespielt?

## Erkenntnis

Keyword-Anfragen und Suchmaschinen reichen hier einfach nicht mehr aus!

- 1 Motivation
- 2 Suchmaschinen
  - Swoogle
  - Falcons
  - Sindice
  - Sig.ma
  - Indexierung
- 3 Lookup-basierte Anfragebearbeitung**
  - Explorative Anfragebearbeitung
  - Index-basierte Anfragebearbeitung
- 4 Systeme zur Anfragebearbeitung in heterogenen Datenquellen
  - Verteilte Datenbanken
  - Naive verteilte Anfragebearbeitung
  - Anfrageoptimierung
- 5 Zusammenfassung

# Explorative Anfragebearbeitung

## Eigenschaften

- Dereferenzieren von URIs
- Keine Statistiken benötigt
- Daten werden während der Anfragebearbeitung heruntergeladen (Live Lookups)

## Basisalgorithmus

- Abwechselnd
  - Teile der Anfrage evaluieren
  - Dereferenziere URIs im Zwischenergebnis (Daten herunterladen)
  - Benutze heruntergeladene Daten zur weiteren Bearbeitung der Anfrage



# Explorative Anfragebearbeitung

## Eigenschaften

- Dereferenzieren von URIs
- Keine Statistiken benötigt
- Daten werden während der Anfragebearbeitung heruntergeladen (Live Lookups)

## Basisalgorithmus

- Abwechselnd
  - Teile der Anfrage evaluieren
  - Dereferenziere URIs im Zwischenergebnis (Daten herunterladen)
  - Benutze heruntergeladene Daten zur weiteren Bearbeitung der Anfrage

# Explorative Anfragebearbeitung

Join-Berechnung durch Nested-Loop (Pull/Iterator-basierte Joins)

## Beispielanfrage

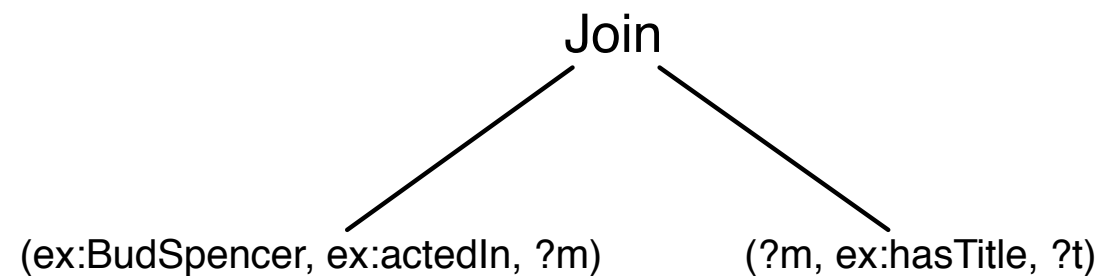
```
SELECT ?m ?t WHERE {  
  ex:BudSpencer ex:actedIn ?m .  
  ?m ex:hasTitle ?t .  
}
```

# Explorative Anfragebearbeitung

Join-Berechnung durch Nested-Loop (Pull/Iterator-basierte Joins)

## Beispielanfrage

```
SELECT ?m ?t WHERE {  
  ex:BudSpencer ex:actedIn ?m .  
  ?m ex:hasTitle ?t .  
}
```

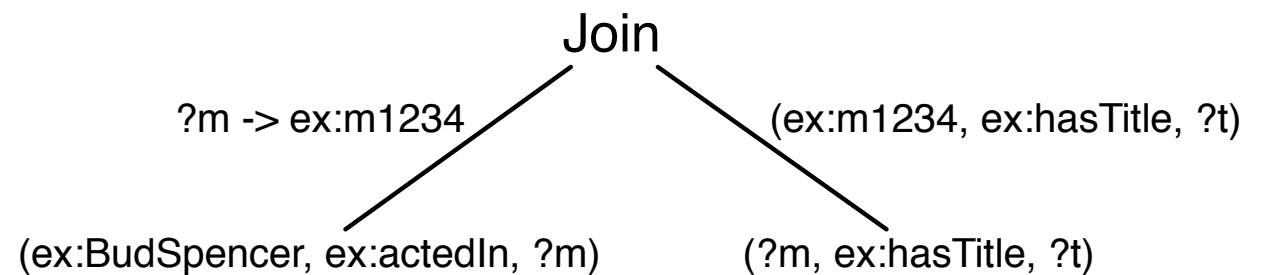


# Explorative Anfragebearbeitung

## Join-Berechnung durch Nested-Loop (Pull/Iterator-basierte Joins)

### Beispielanfrage

```
SELECT ?m ?t WHERE {  
  ex:BudSpencer ex:actedIn ?m .  
  ?m ex:hasTitle ?t .  
}
```

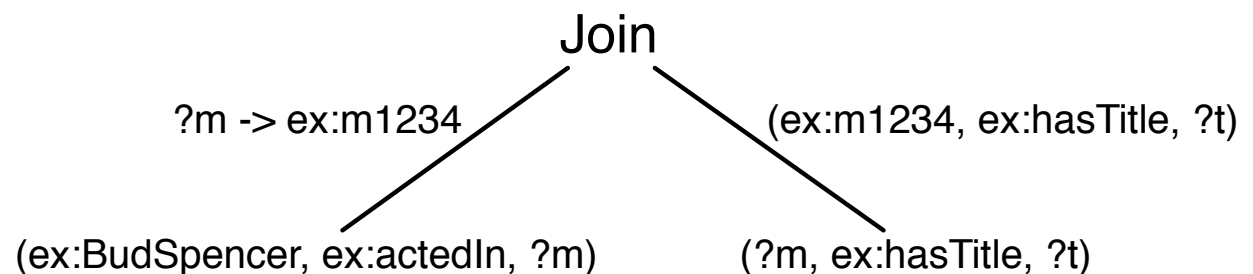


# Explorative Anfragebearbeitung

## Join-Berechnung durch Nested-Loop (Pull/Iterator-basierte Joins)

### Beispielanfrage

```
SELECT ?m ?t WHERE {  
  ex:BudSpencer ex:actedIn ?m .  
  ?m ex:hasTitle ?t .  
}
```



Problem, wenn die Daten für “ex:m1234” noch nicht heruntergeladen wurden

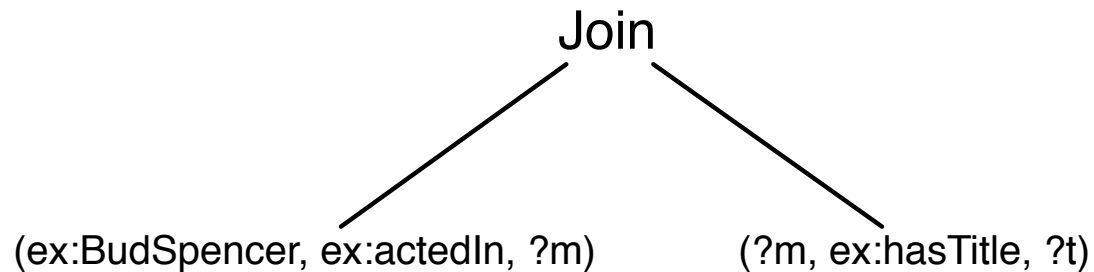
- Join-Operator wartet bis die Daten verfügbar sind (Blocking)
- Non-Blocking Iterator Join [HBF09]
  - Vorläufiges Zurückweisen von Join-Input der unteren Ebenen
  - Neuer Versuch später
- Reihenfolge der Dereferenzierung beeinflusst Vollständigkeit [Har11]



# Explorative Anfragebearbeitung

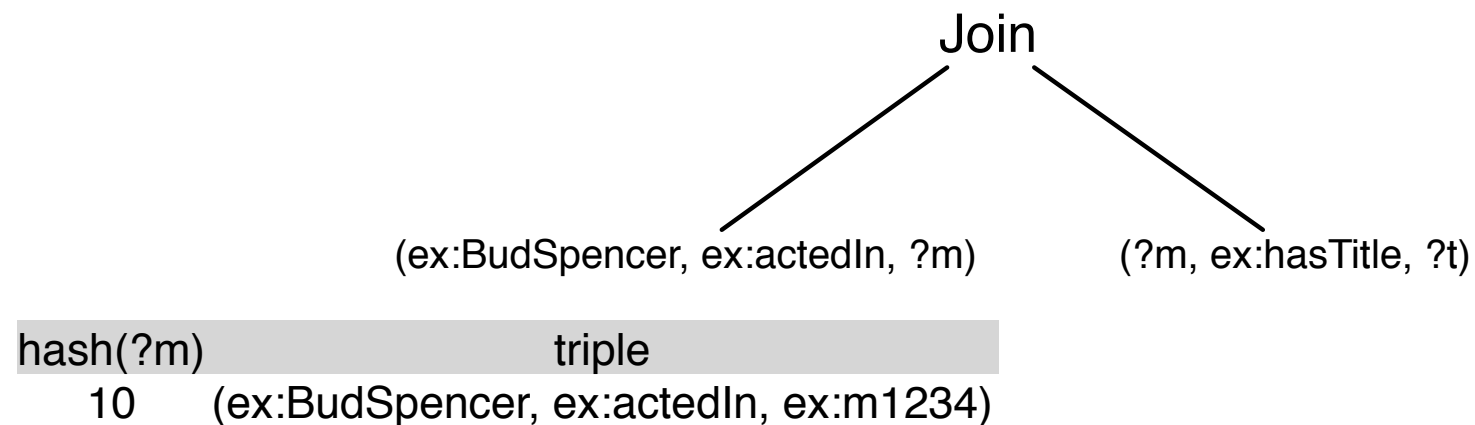
## Symmetrischer Hash Join über Linked Data [LT10]

- Push-basiert statt pull-basiert
- Vermeidung von Wartezeiten
- Effizientere Laufzeit



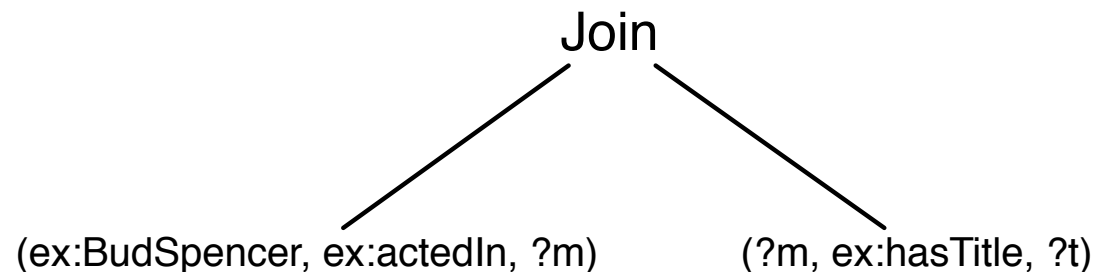
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



# Explorative Anfragebearbeitung

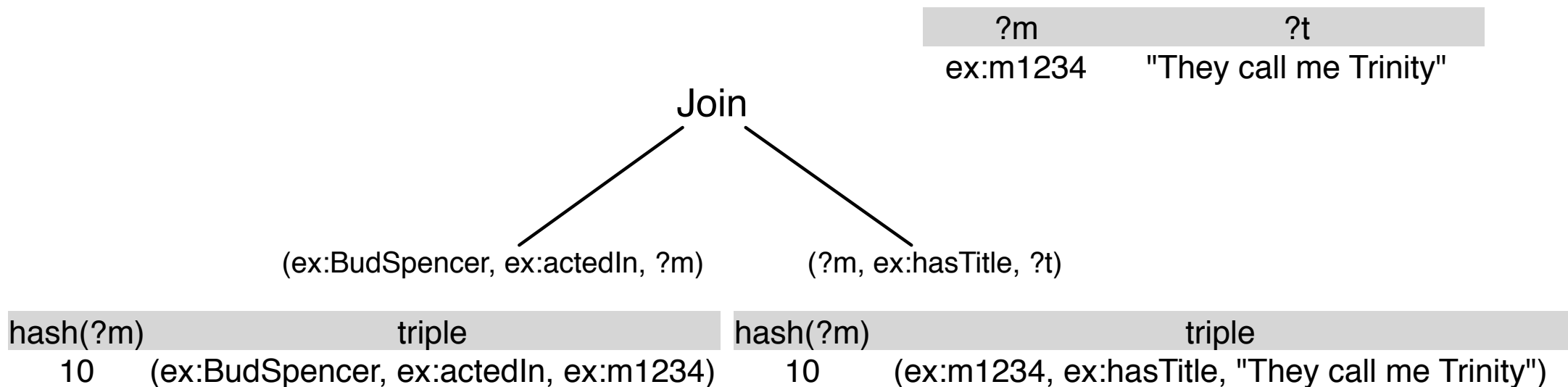
## Symmetrischer Hash Join über Linked Data [LT10]



| hash(?m) | triple                                | hash(?m) | triple  |
|----------|---------------------------------------|----------|---|
| 10       | (ex:BudSpencer, ex:actedIn, ex:m1234) | 10       | (ex:m1234, ex:hasTitle, "They call me Trinity") |

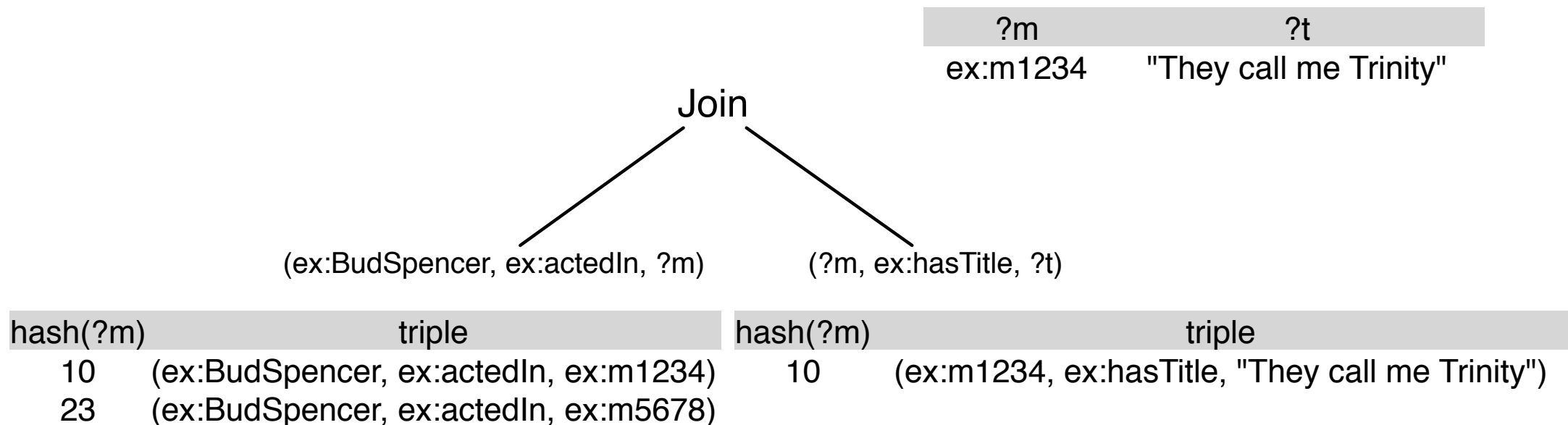
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



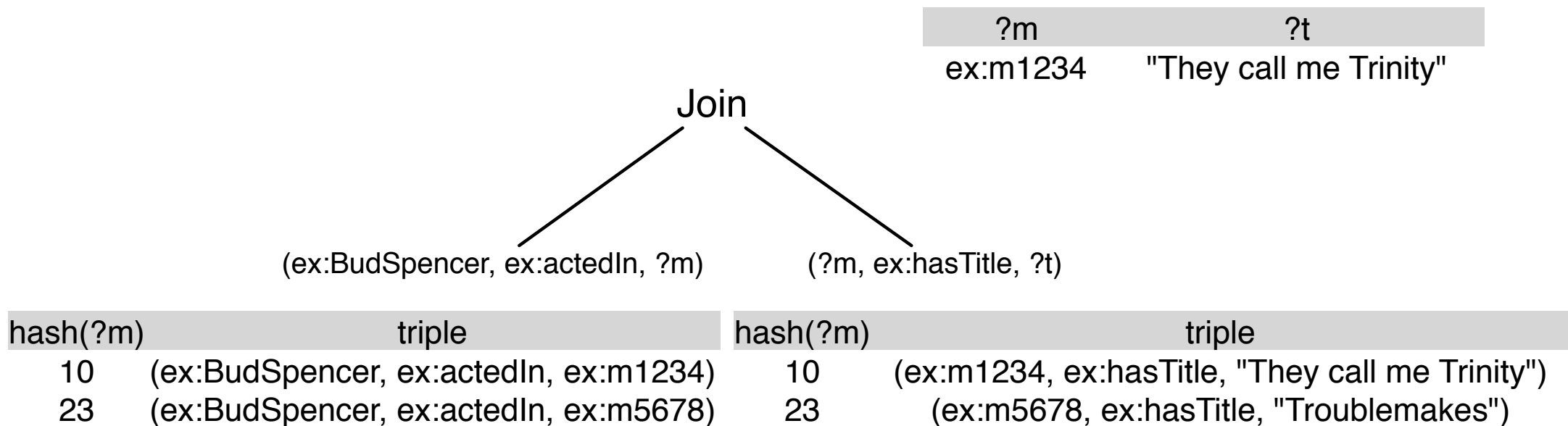
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



# Explorative Anfragebearbeitung

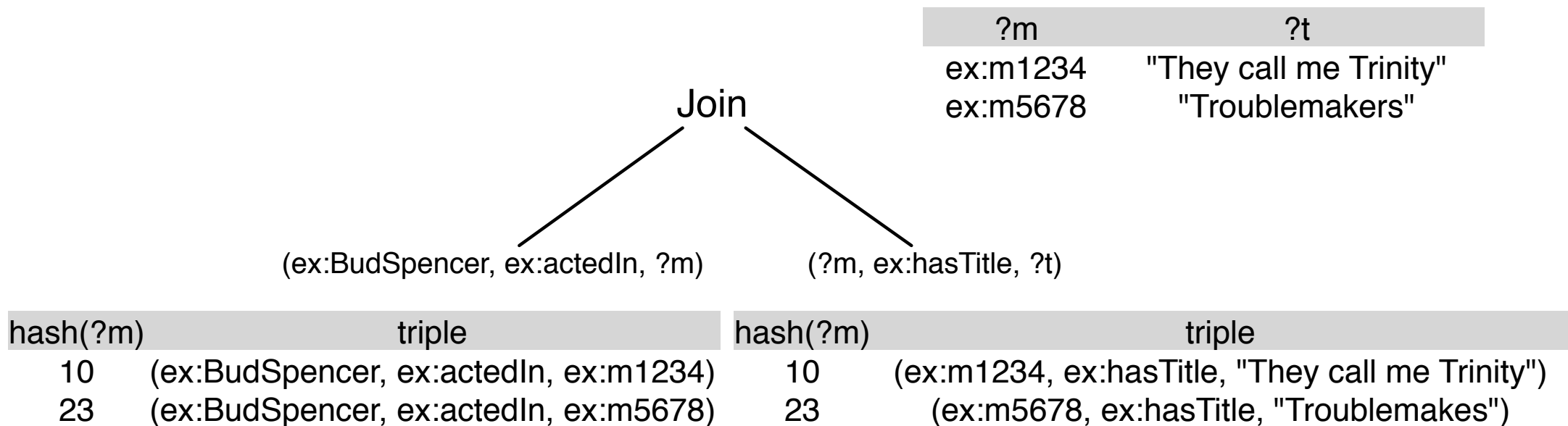
## Symmetrischer Hash Join über Linked Data [LT10]





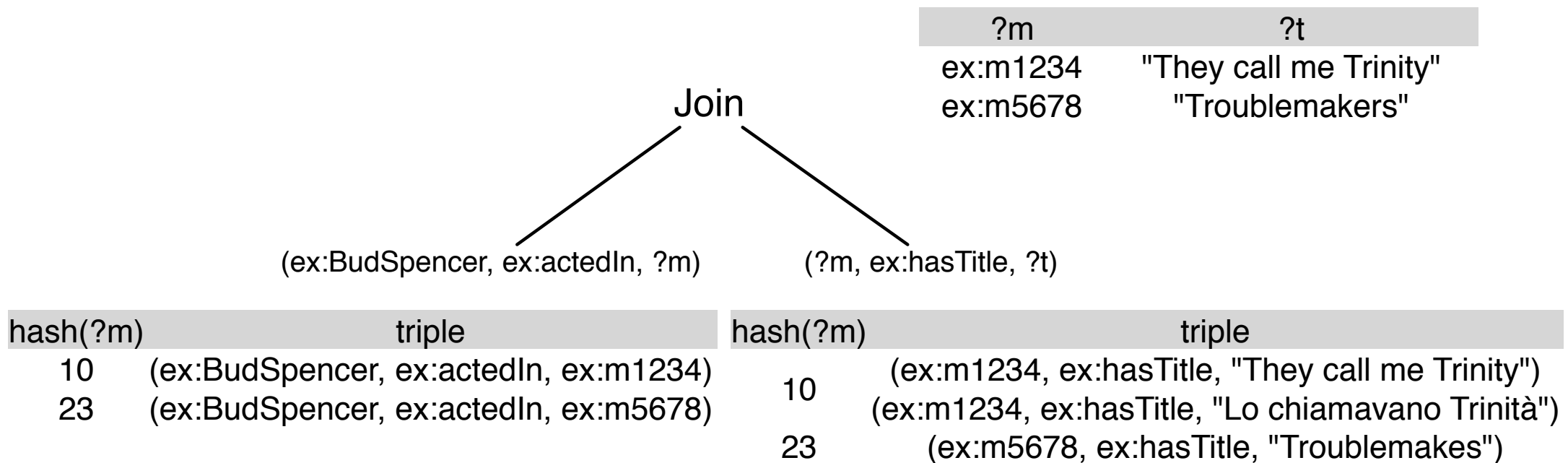
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



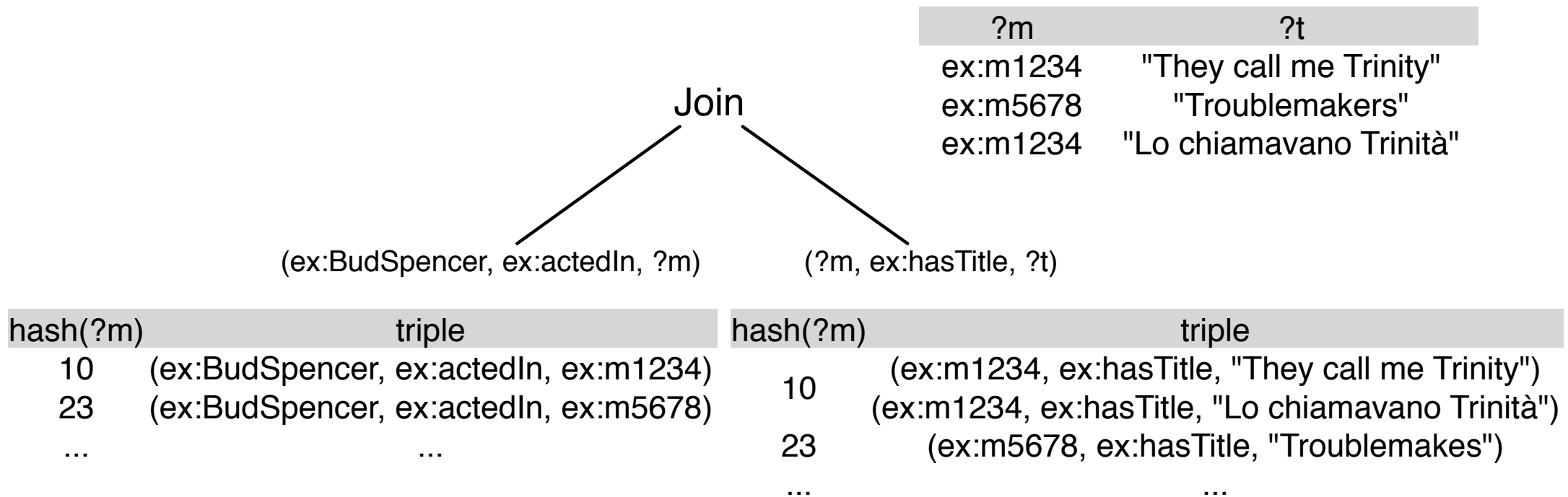
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



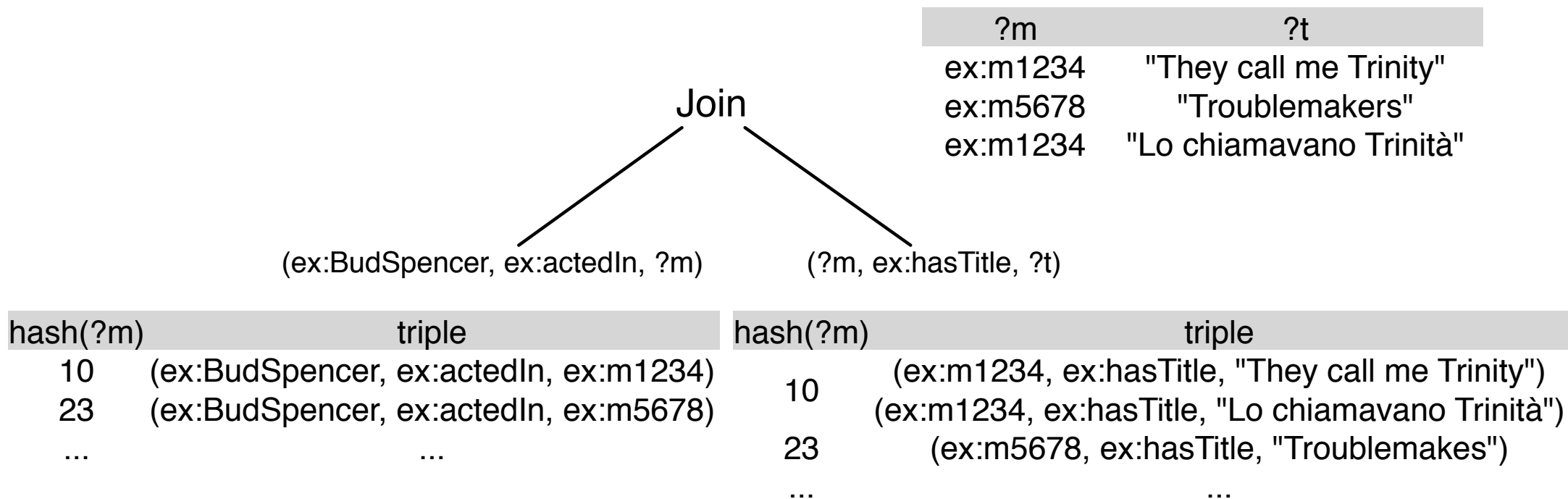
# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



# Explorative Anfragebearbeitung

## Symmetrischer Hash Join über Linked Data [LT10]



## Erweiterung für die Kombination mit lokalen Daten [LT11]

- Einige Daten liegen lokal vor
- Indexe über lokale Daten zur Effizienzsteigerung

# Index-basierte Anfragebearbeitung

## Eigenschaften

- Potentielle Quellen im Voraus indexieren (Crawling, Dereferencing URIs, etc.)
- Relevante Quellen für eine Anfrage identifizieren
- Daten der Quellen parallel herunterladen

## Erweiterter Index [UHK<sup>+</sup>11]

- Indiziere (S,P,O)-Kombination
- Basis: multidimensionale Histogramme
- Identifikation relevanter Quellen für Triple-Patterns
- Identifikation relevanter Quellen für Joins

# Index-basierte Anfragebearbeitung

## Eigenschaften

- Potentielle Quellen im Voraus indexieren (Crawling, Dereferencing URIs, etc.)
- Relevante Quellen für eine Anfrage identifizieren
- Daten der Quellen parallel herunterladen

## Erweiterter Index [UHK<sup>+</sup>11]

- Indiziere (S,P,O)-Kombination
- Basis: multidimensionale Histogramme
- Identifikation relevanter Quellen für Triple-Patterns
- Identifikation relevanter Quellen für Joins



# Index-basierte Anfragebearbeitung

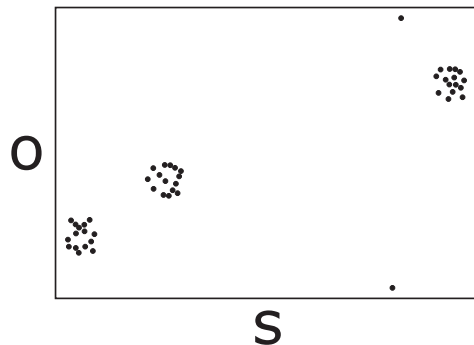
## Histogramme

- Transformiere Triple in numerischen Raum (Hash-Funktionen)  
(`ex:BudSpencer ex:actedIn ex:m1234`)  $\rightarrow$  (323, 232, 124)
- In entsprechendes Bucket einfügen
- Lookup: Transformiere Triple-Pattern in numerischen Raum  
(`ex:BudSpencer ex:actedIn ?m`)

# Index-basierte Anfragebearbeitung

## Histogramme

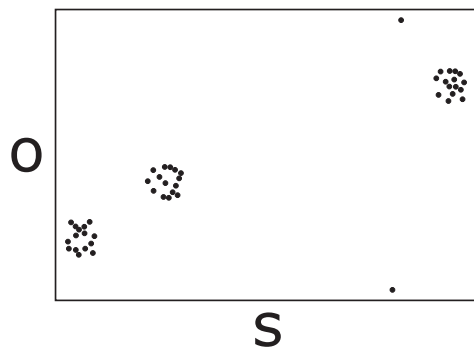
- Transformiere Triple in numerischen Raum (Hash-Funktionen)  
(`ex:BudSpencer ex:actedIn ex:m1234`)  $\rightarrow$  (323, 232, 124)
- In entsprechendes Bucket einfügen
- Lookup: Transformiere Triple-Pattern in numerischen Raum  
(`ex:BudSpencer ex:actedIn ?m`)



# Index-basierte Anfragebearbeitung

## Histogramme

- Transformiere Triple in numerischen Raum (Hash-Funktionen)  
(`ex:BudSpencer ex:actedIn ex:m1234`)  $\rightarrow$  (323, 232, 124)
- In entsprechendes Bucket einfügen
- Lookup: Transformiere Triple-Pattern in numerischen Raum  
(`ex:BudSpencer ex:actedIn ?m`)

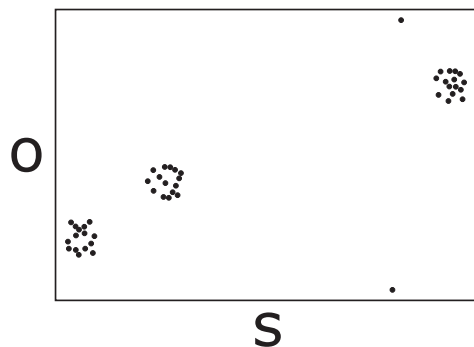


|   |    |   |    |
|---|----|---|----|
|   | 0  | 0 | 16 |
| 0 | 15 | 0 | 0  |
|   | 15 | 0 | 1  |
|   | S  |   |    |

# Index-basierte Anfragebearbeitung

## Histogramme

- Transformiere Triple in numerischen Raum (Hash-Funktionen)  
(`ex:BudSpencer ex:actedIn ex:m1234`)  $\rightarrow$  (323, 232, 124)
- In entsprechendes Bucket einfügen
- Lookup: Transformiere Triple-Pattern in numerischen Raum  
(`ex:BudSpencer ex:actedIn ?m`)



|   |    |   |    |
|---|----|---|----|
| o | 0  | 0 | 16 |
|   | 15 | 0 | 0  |
|   | 15 | 0 | 1  |
|   | S  |   |    |

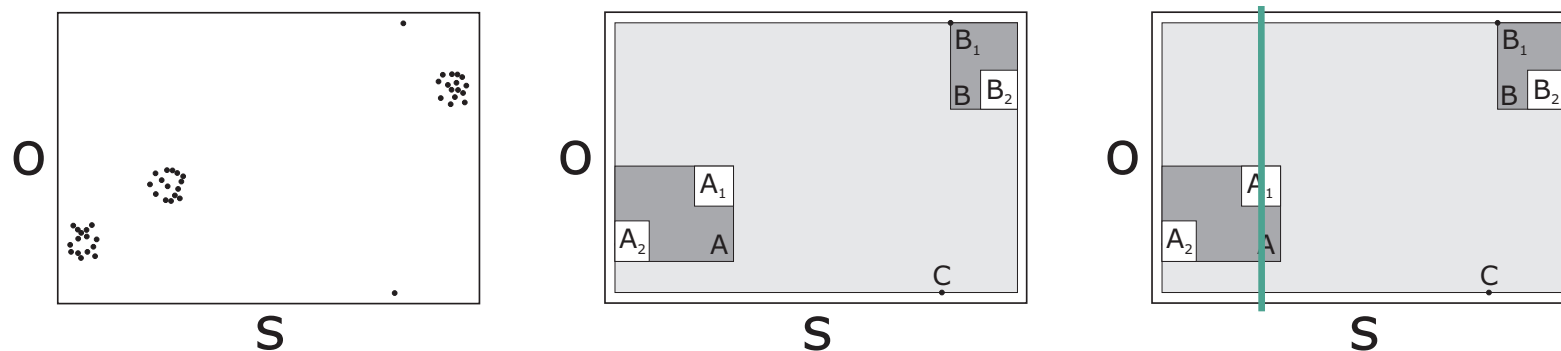
|   |    |   |    |
|---|----|---|----|
| o | 0  | 0 | 16 |
|   | 15 | 0 | 0  |
|   | 15 | 0 | 1  |
|   | S  |   |    |

# Index-basierte Anfragebearbeitung

## Histogramme

- Transformiere Triple in numerischen Raum (Hash-Funktionen)  
(`ex:BudSpencer ex:actedIn ex:m1234`)  $\rightarrow$  (323, 232, 124)
- In entsprechendes Bucket einfügen
- Lookup: Transformiere Triple-Pattern in numerischen Raum  
(`ex:BudSpencer ex:actedIn ?m`)

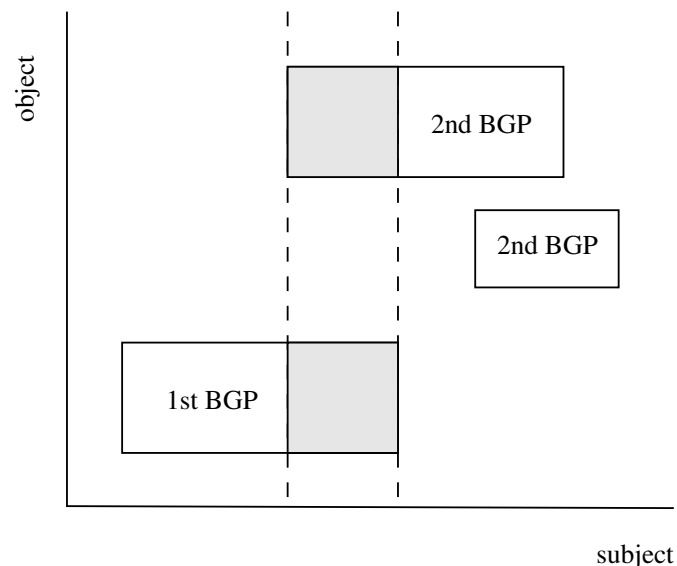
Alternative zu Histogrammen (QTree oder Clustering) [UHK<sup>+</sup>11]:



# Index-basierte Anfragebearbeitung

## Kardinalitätsabschätzung für Joins

- Buckets für 1. Triple-Pattern ermitteln
- Buckets für 2. Triple-Pattern ermitteln
- Buckets bestimmen, die in der Join-Dimension überlappen (z.B. subject)
- Schätze Ergebniskardinalität durch Überlappungsgrad



# Lokale Anfragebearbeitung

## Pro und Kontra

- Live-Lookups garantieren Aktualität
- Sehr viel Network-Traffic
- Alles wird von einer zentralen Instanz (Koordinator) gemacht

Warum nicht die Rechenleistung der SPARQL-Endpoints nutzen?

*Verteilte Anfragebearbeitung*



# Lokale Anfragebearbeitung

## Pro und Kontra

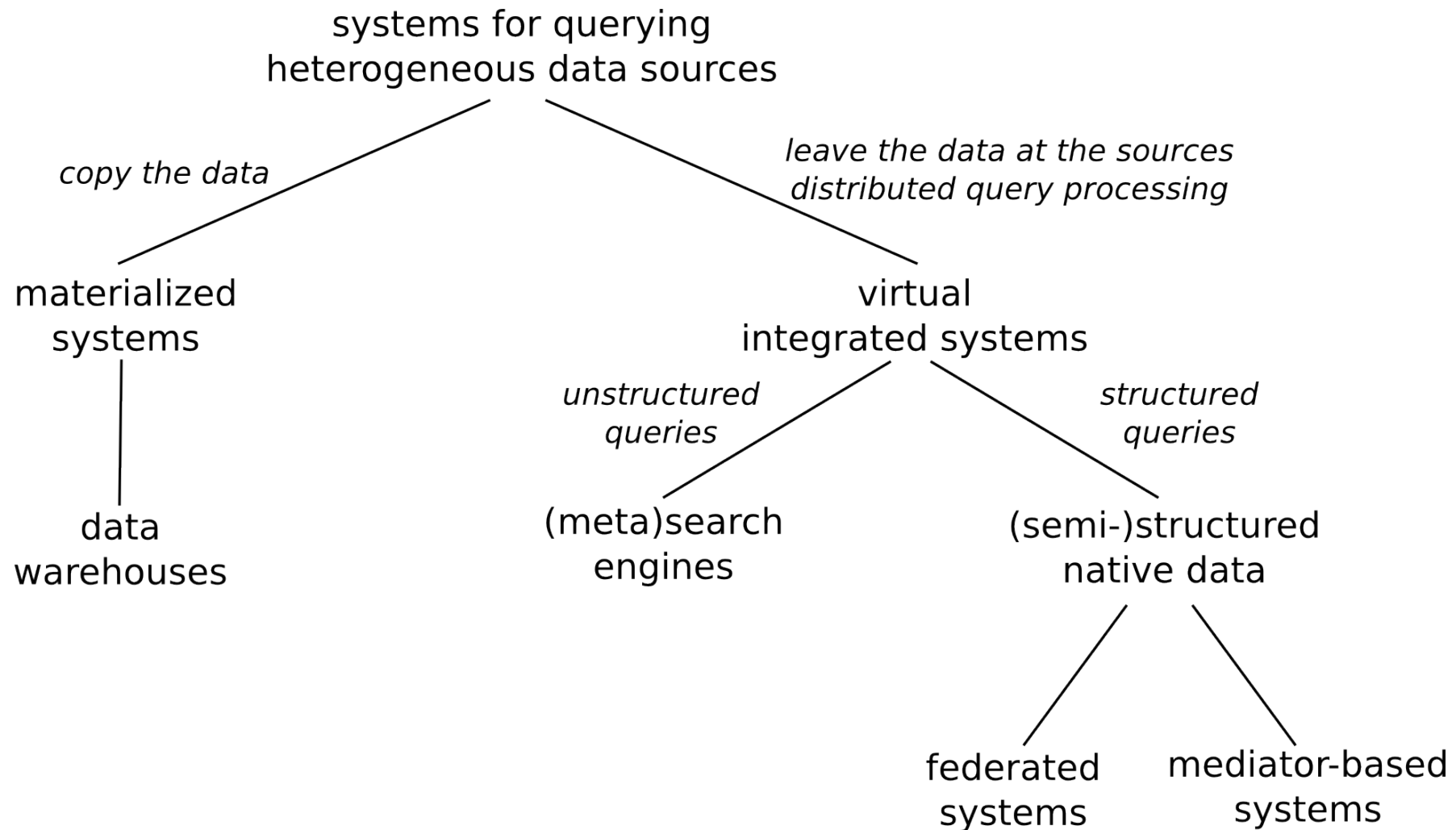
- Live-Lookups garantieren Aktualität
- Sehr viel Network-Traffic
- Alles wird von einer zentralen Instanz (Koordinator) gemacht

Warum nicht die Rechenleistung der SPARQL-Endpoints nutzen?

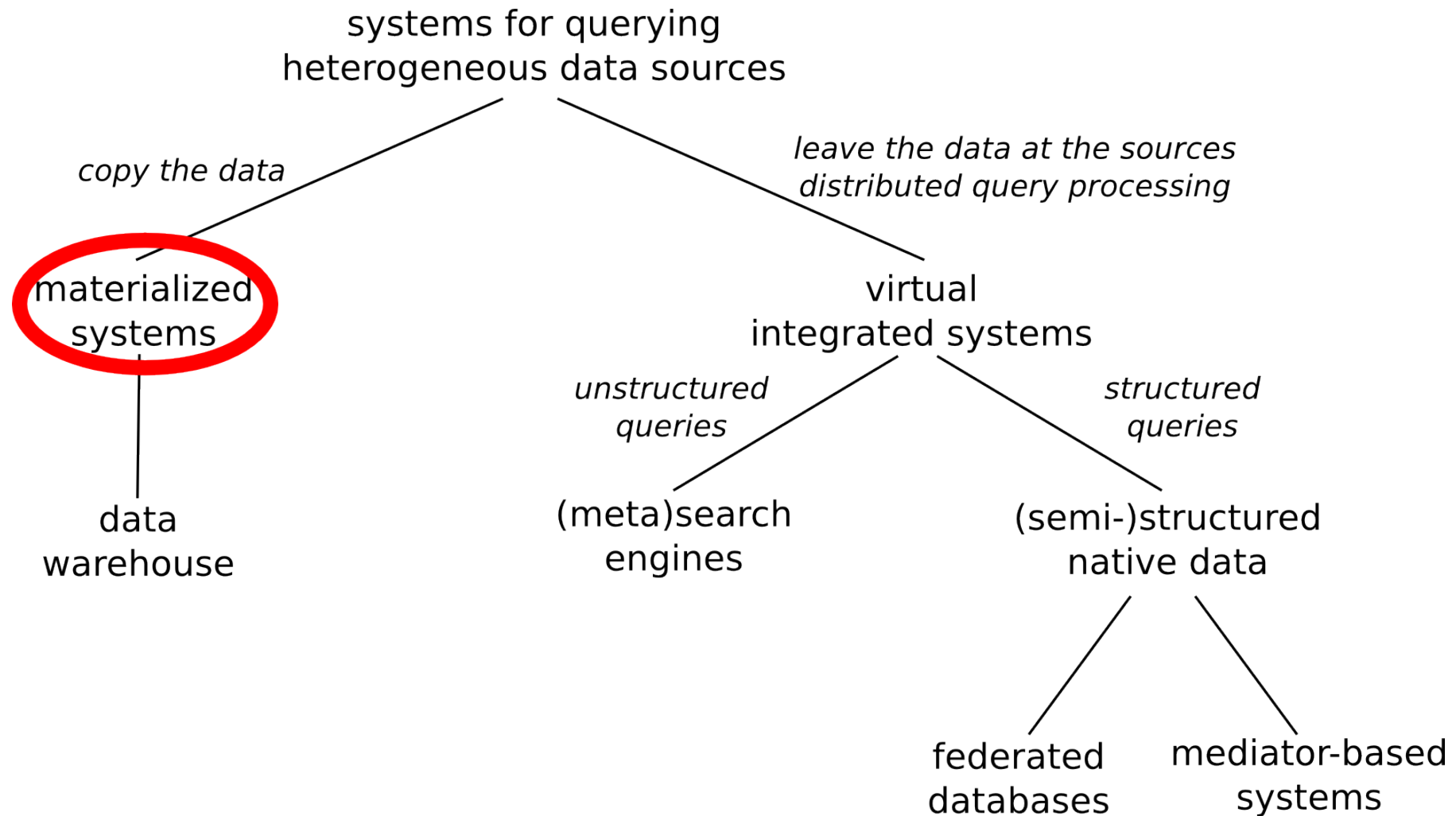
*Verteilte Anfragebearbeitung*

- 1 Motivation
- 2 Suchmaschinen
  - Swoogle
  - Falcons
  - Sindice
  - Sig.ma
  - Indexierung
- 3 Lookup-basierte Anfragebearbeitung
  - Explorative Anfragebearbeitung
  - Index-basierte Anfragebearbeitung
- 4 Systeme zur Anfragebearbeitung in heterogenen Datenquellen
  - Verteilte Datenbanken
  - Naive verteilte Anfragebearbeitung
  - Anfrageoptimierung
- 5 Zusammenfassung

# Datenintegrationssysteme [DD99]



# Datenintegrationssysteme [DD99]



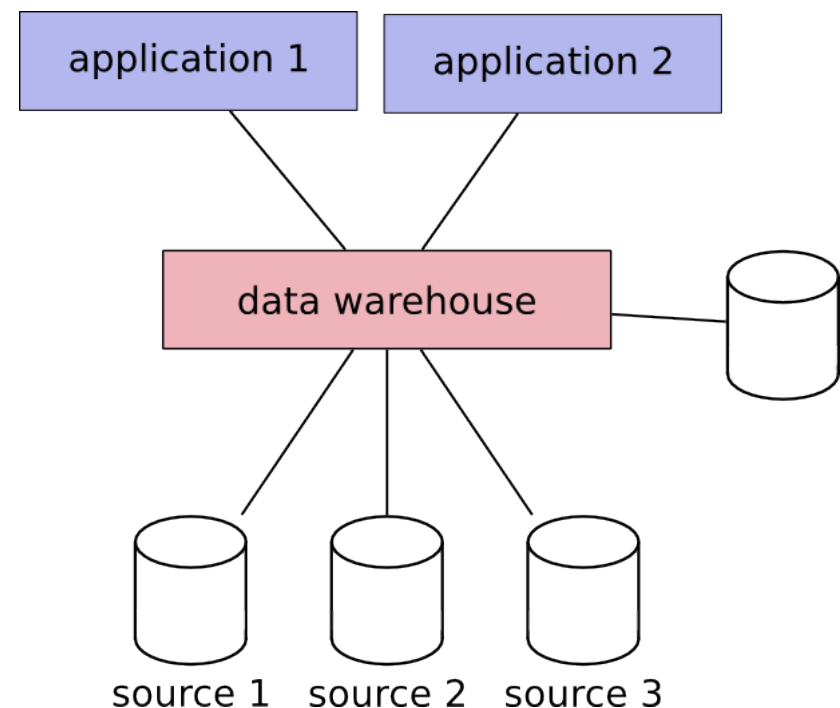
# Data Warehousing

## Datenfluss

- ETL-Prozess
  - Extraktion
  - Transformation
  - Laden
- Redundante Datenhaltung
- Periodischer Datenimport
- Aktualität

## Anfragebearbeitung

- Zentrale Anfragebearbeitung mit lokaler Optimierung



Funktioniert für RDF und Linked Data im Prinzip genauso

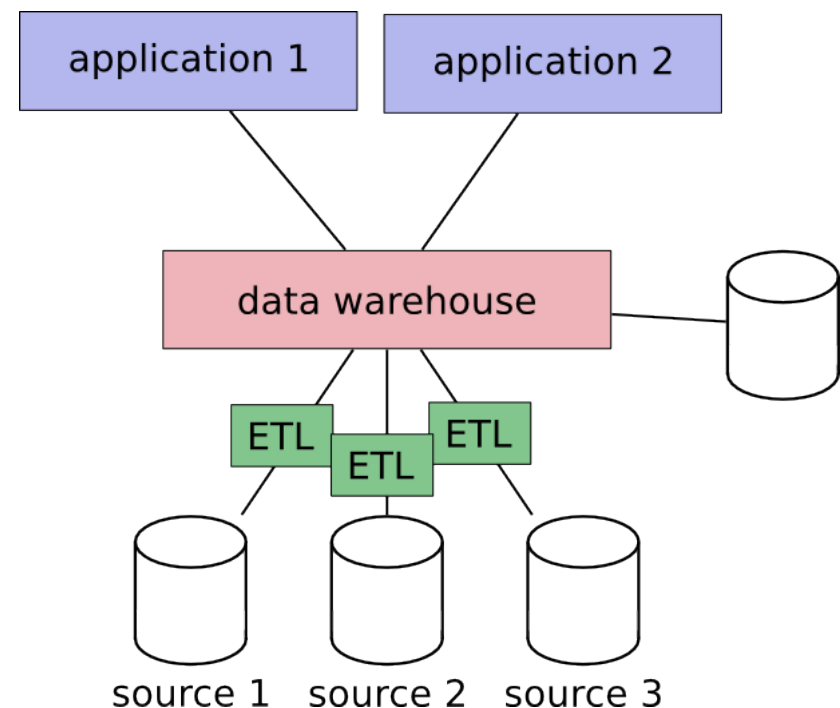
# Data Warehousing

## Datenfluss

- ETL-Prozess
  - Extraktion
  - Transformation
  - Laden
- Redundante Datenhaltung
- Periodischer Datenimport
- Aktualität

## Anfragebearbeitung

- Zentrale Anfragebearbeitung mit lokaler Optimierung



Funktioniert für RDF und Linked Data im Prinzip genauso

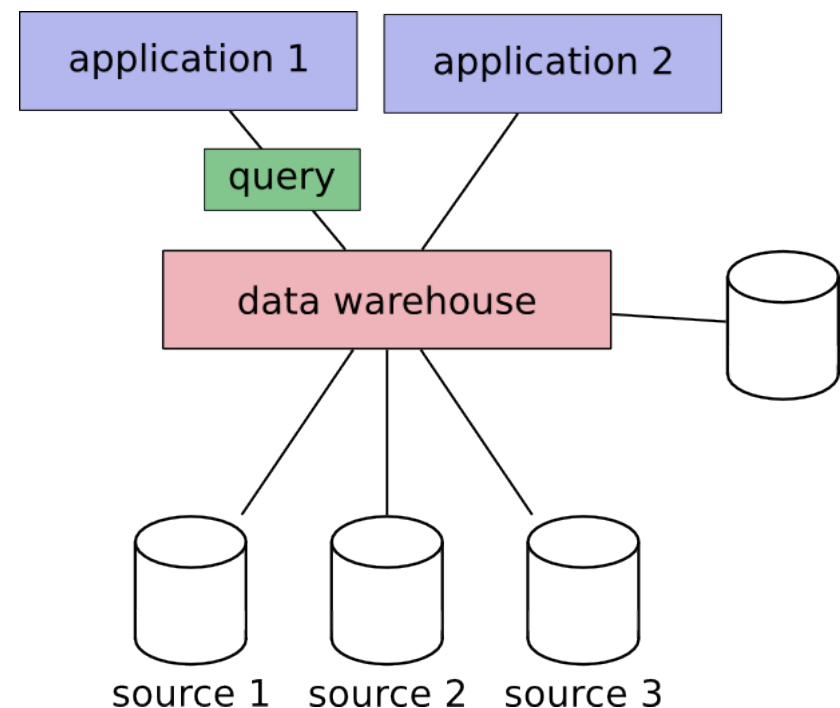
# Data Warehousing

## Datenfluss

- ETL-Prozess
  - Extraktion
  - Transformation
  - Laden
- Redundante Datenhaltung
- Periodischer Datenimport
- Aktualität

## Anfragebearbeitung

- Zentrale Anfragebearbeitung mit lokaler Optimierung



Funktioniert für RDF und Linked Data im Prinzip genauso



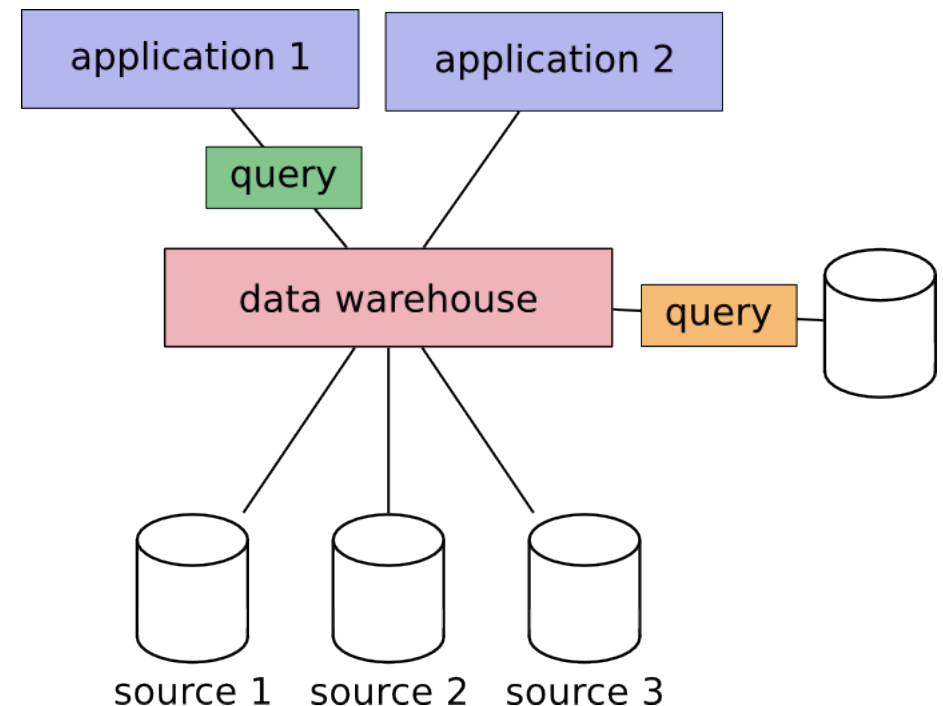
# Data Warehousing

## Datenfluss

- ETL-Prozess
  - Extraktion
  - Transformation
  - Laden
- Redundante Datenhaltung
- Periodischer Datenimport
- Aktualität

## Anfragebearbeitung

- Zentrale Anfragebearbeitung mit lokaler Optimierung



Funktioniert für RDF und Linked Data im Prinzip genauso

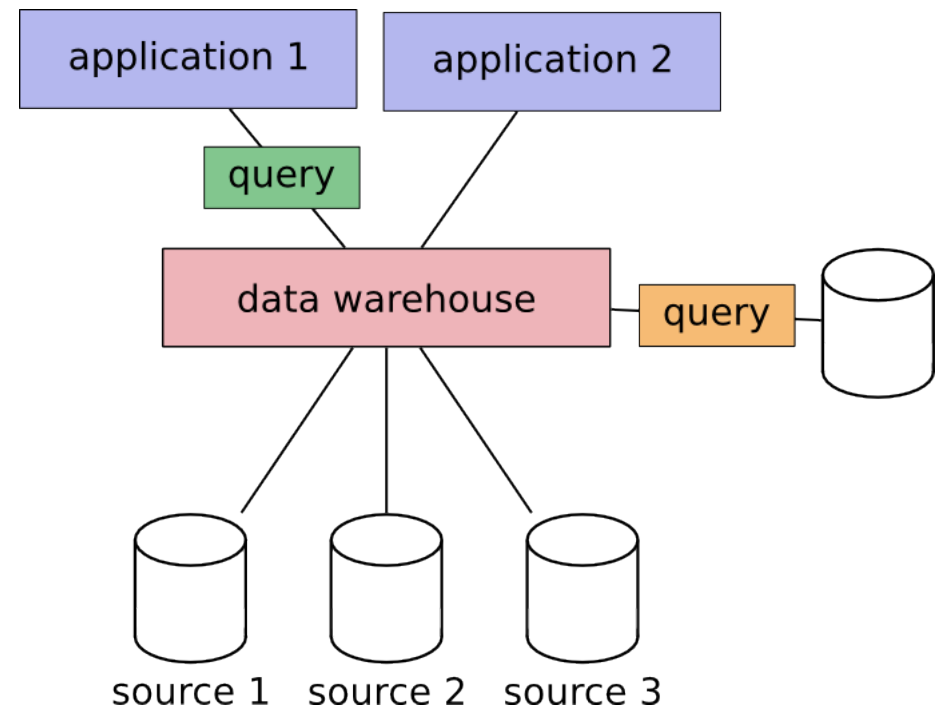
# Data Warehousing

## Datenfluss

- ETL-Prozess
  - Extraktion
  - Transformation
  - Laden
- Redundante Datenhaltung
- Periodischer Datenimport
- Aktualität

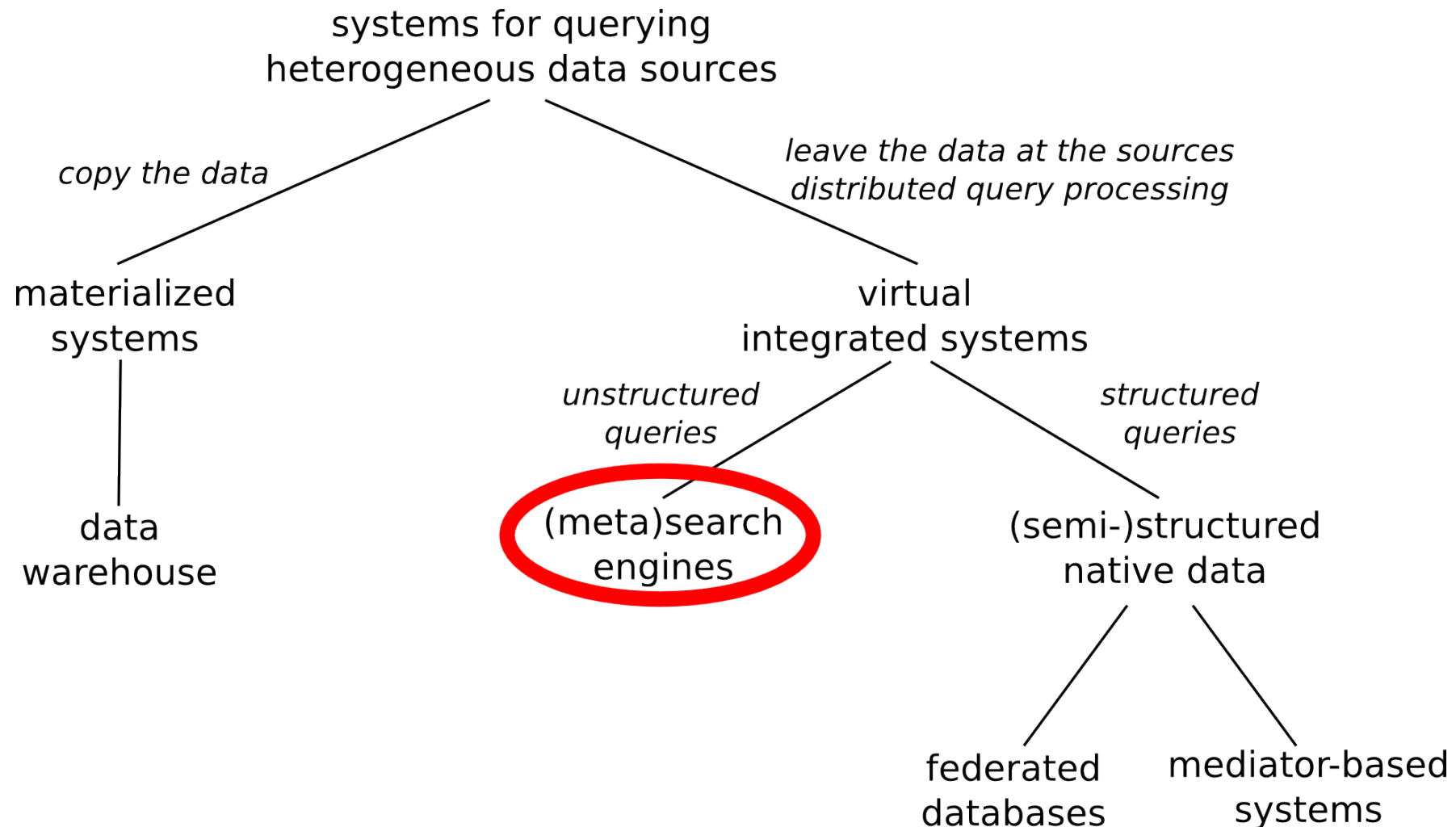
## Anfragebearbeitung

- Zentrale Anfragebearbeitung mit lokaler Optimierung

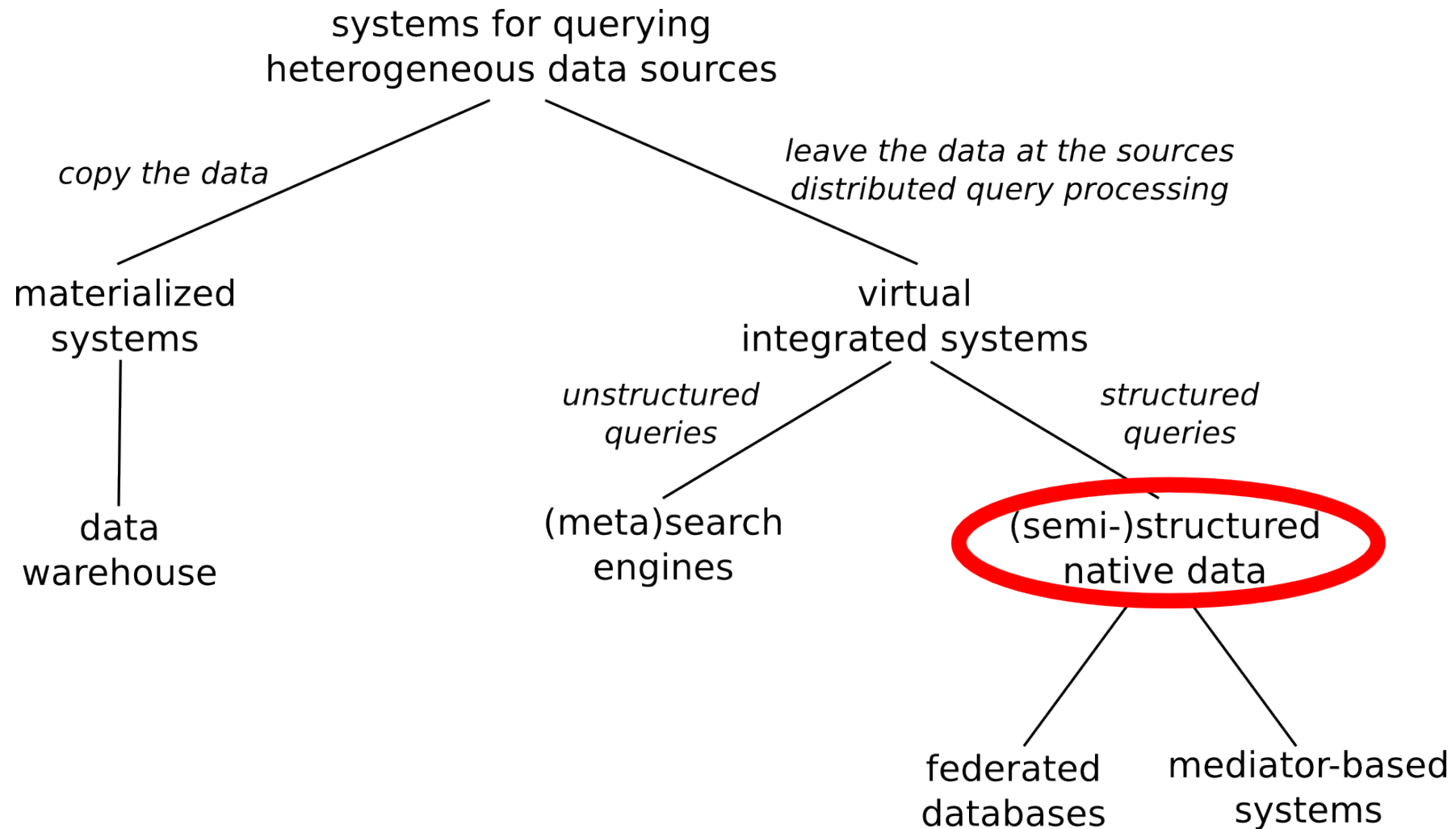


Funktioniert für RDF und Linked Data im Prinzip genauso

# Datenintegrationssysteme [DD99]



# Datenintegrationssysteme [DD99]



# Föderierte und Mediator-Wrapper-Systeme

## Gemeinsamkeiten

- Zusammenschluss eigenständiger Datenquellen
- Ein globales Interface

## Föderierte Systeme

- Aktive Unterstützung und Kooperation
- Datenquellen überwinden Heterogenität (Datenmodell, Anfragesprache, etc.)

## Mediator-Wrapper-Systeme

- Mediator hat Statistiken
- Wrapper überwindet Heterogenität
- Keine zusätzliche Kooperation der Quellen (Autonomie)

# Föderierte und Mediator-Wrapper-Systeme

## Gemeinsamkeiten

- Zusammenschluss eigenständiger Datenquellen
- Ein globales Interface

## Föderierte Systeme

- Aktive Unterstützung und Kooperation
- Datenquellen überwinden Heterogenität (Datenmodell, Anfragesprache, etc.)

## Mediator-Wrapper-Systeme

- Mediator hat Statistiken
- Wrapper überwindet Heterogenität
- Keine zusätzliche Kooperation der Quellen (Autonomie)

# Föderierte und Mediator-Wrapper-Systeme

## Terminologie in der Semantic-Web-Community

- Virtual integration
- Federated systems
- Federator
- Mediator/wrapper
- Federated query processing
- . . .

# Föderierte und Mediator-Wrapper-Systeme

## Terminologie in der Semantic-Web-Community

- Virtual integration
- Federated systems
- Federator
- Mediator/wrapper
- Federated query processing
- ...



# Föderierte und Mediator-Wrapper-Systeme

## Terminologie in der Semantic-Web-Community

- Virtual integration
- Federated systems
- Federator
- Mediator/wrapper
- Federated query processing
- ...

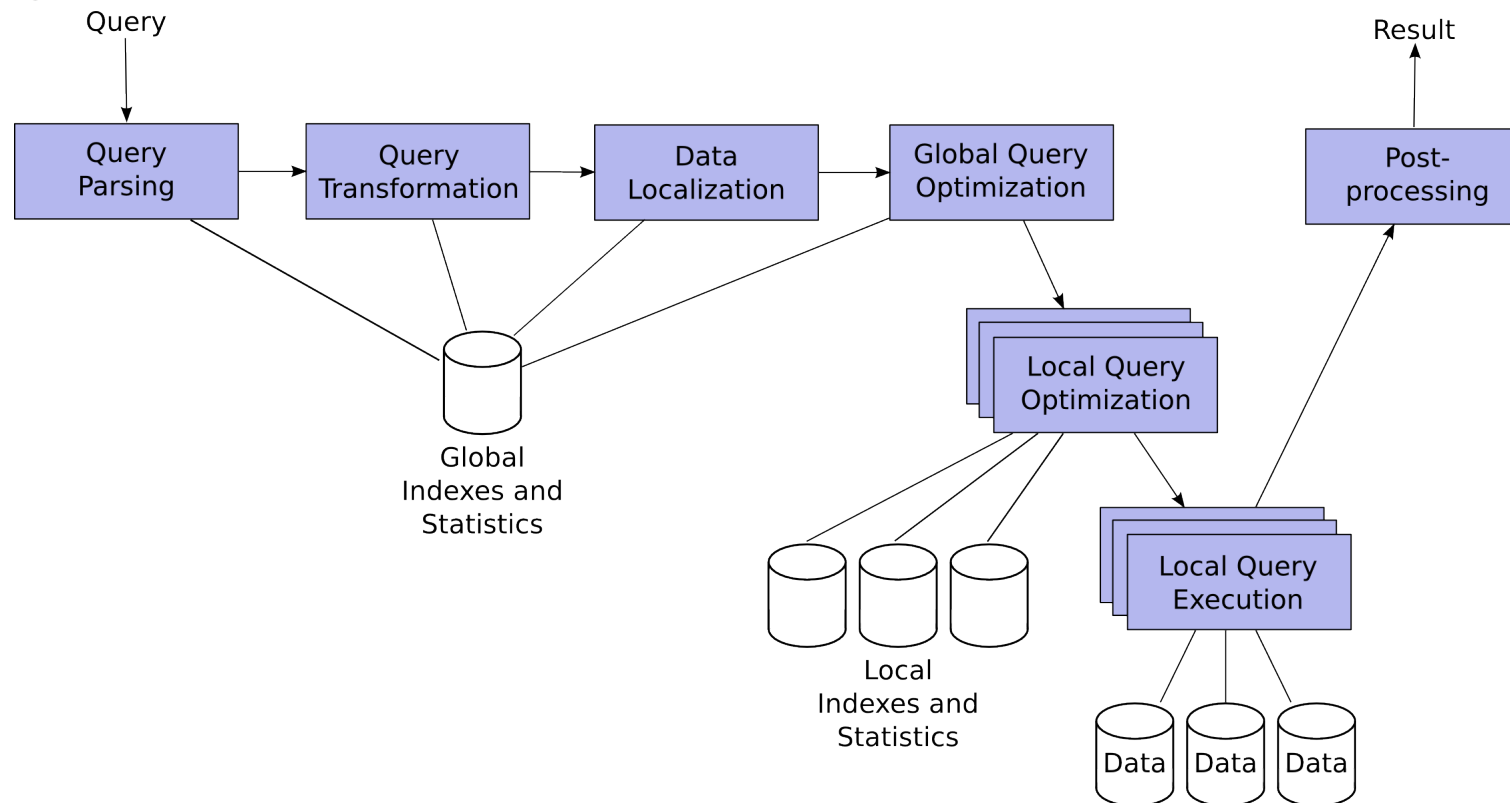
Alles wird mehr oder weniger gleich verwendet

# Verteilte Anfragebearbeitung und Linked-Open-Data

## Verteilte Anfragebearbeitung für Linked-Open-Data

- Entspricht in etwa der virtuellen Datenintegration
- Vieles ähnlich zu verteilten Datenbanken
- Strukturierte Anfragen: SPARQL

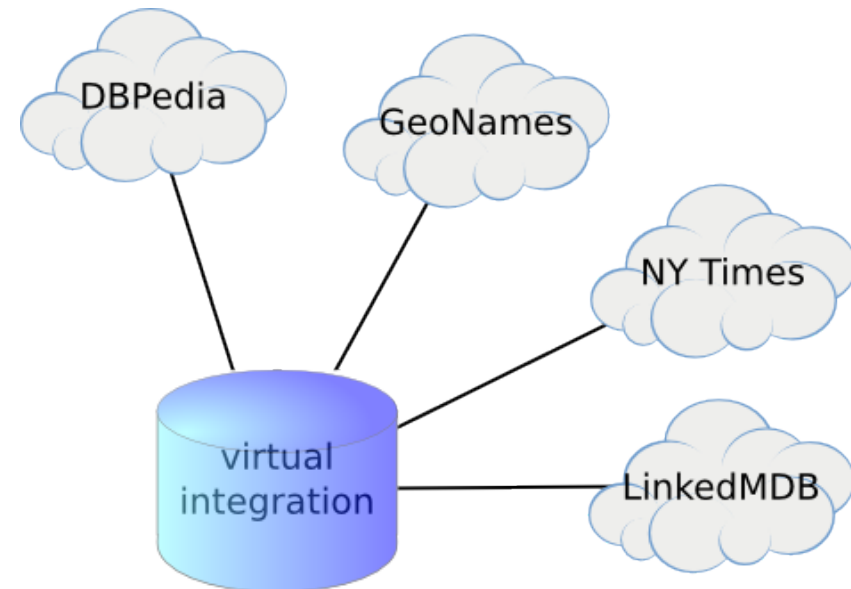
## Grundlegende Schritte



# Naive verteilte Anfragebearbeitung

## Szenario

- SPARQL-Endpoints
- Keine Indexe



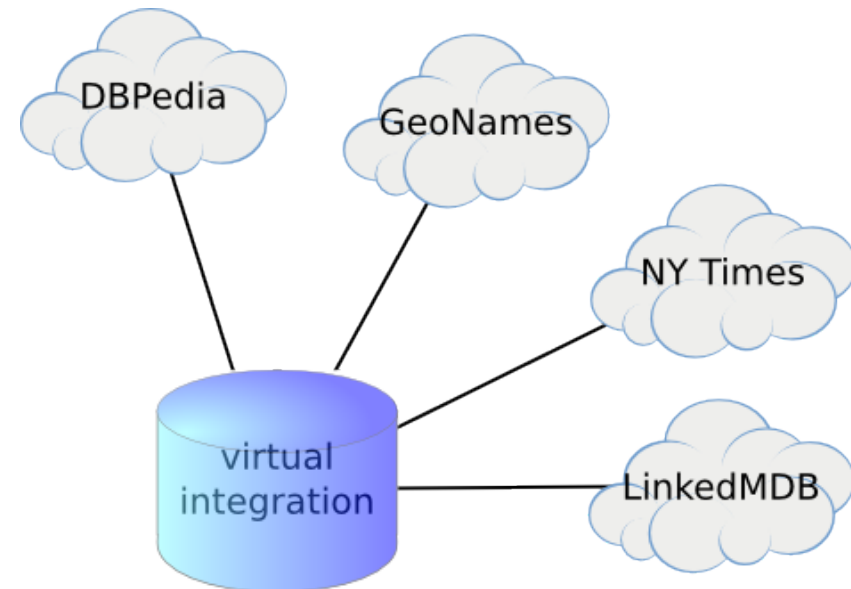
# Naive verteilte Anfragebearbeitung

## Szenario

- SPARQL-Endpoints
- Keine Indexe

## Beispielanfrage

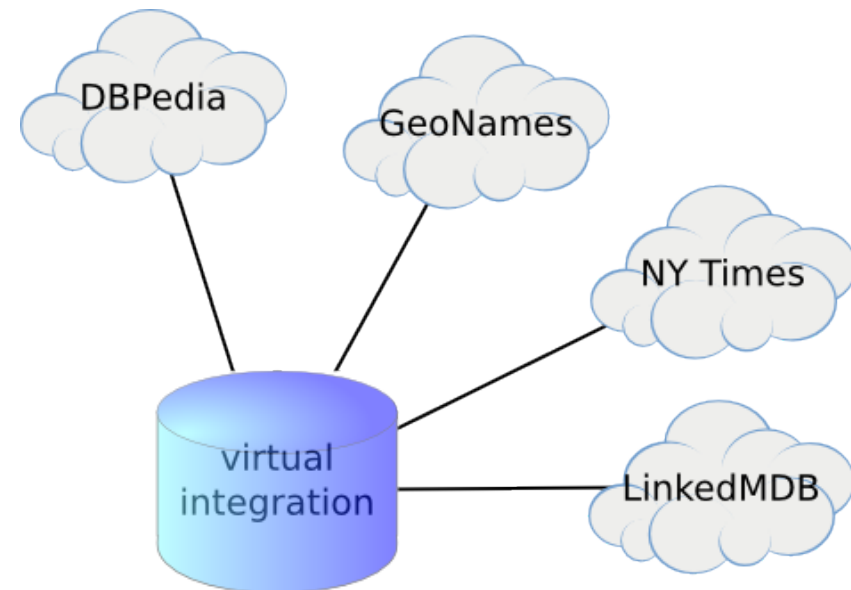
```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```



# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

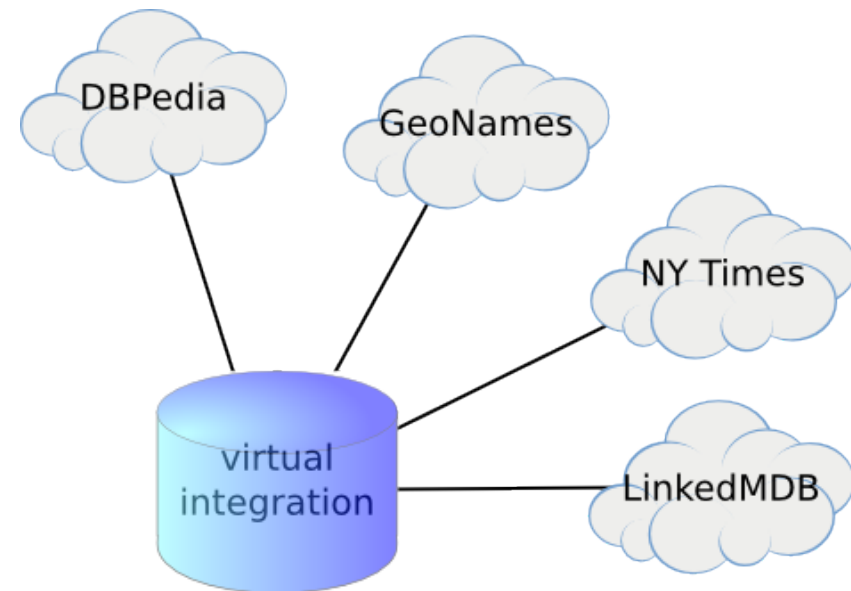


# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

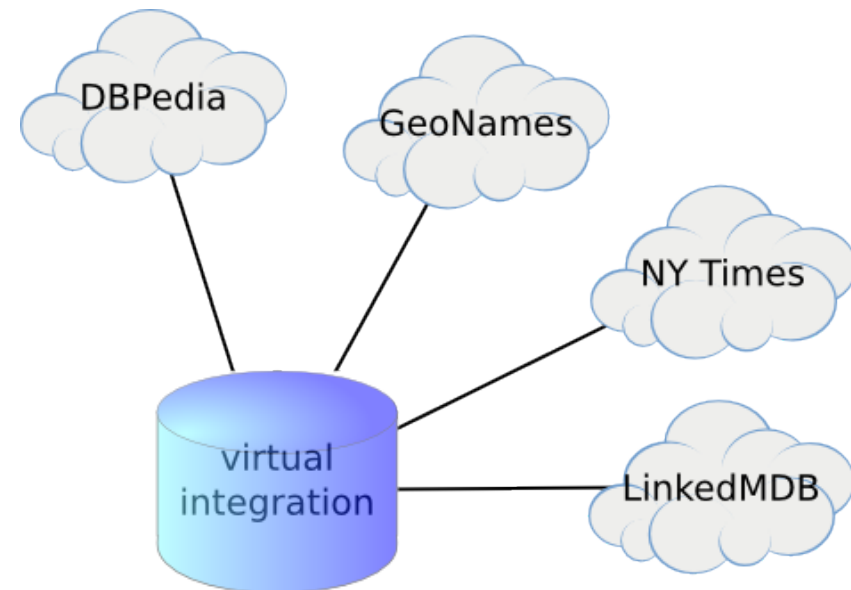
- Sende Nachricht mit Triple-Pattern an alle 4 Quellen  
→ 4 Requests
- Empfange 200 Mappings für ?Country und ?Capital  
e.g., ?Country=ex:Germany,  
?Capital=ex:Berlin



# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

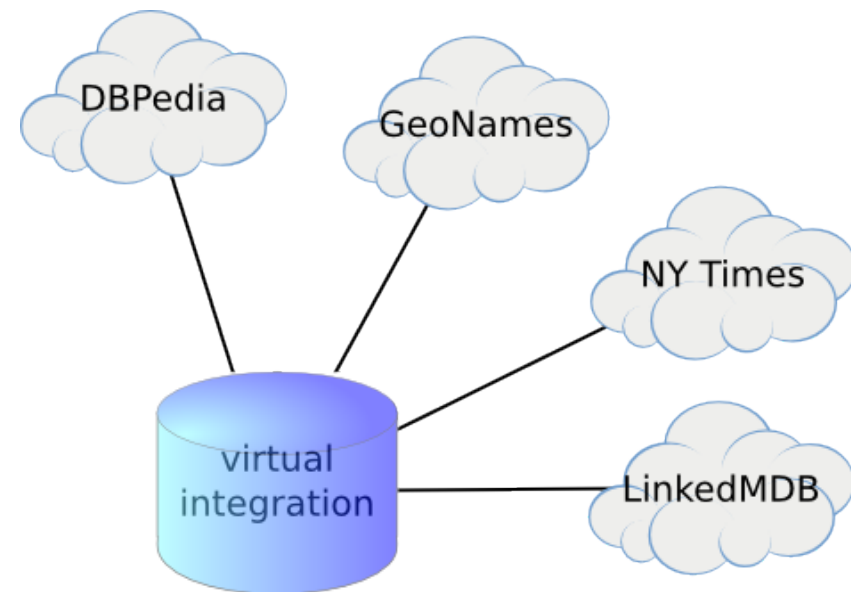


# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

- Benutze Ergebnisse für 2. Triple-Statement (Nested-Loop)
- $200 \times 4$  Requests  
e.g., `SELECT ?CountryPop WHERE {ex:Germany ex:population ?CountryPop .}`
- 150 Mappings

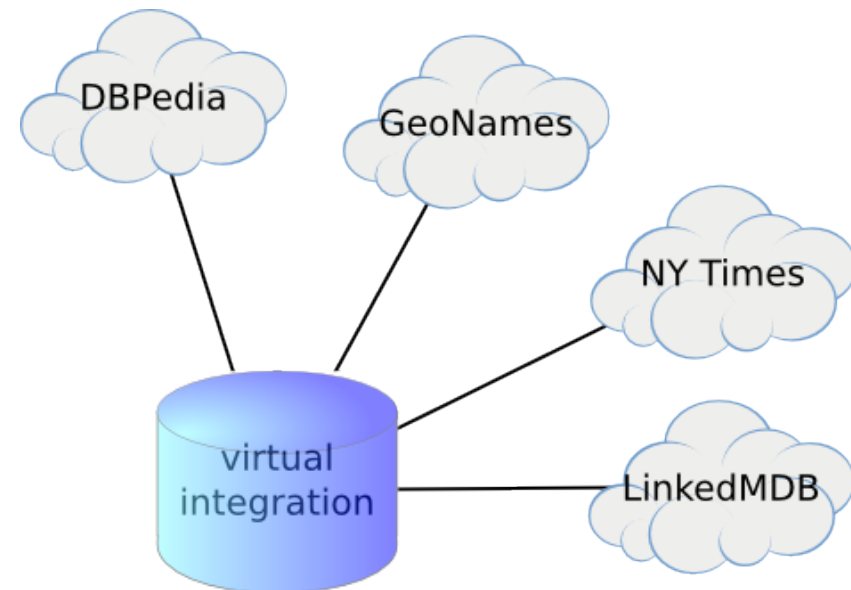




# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

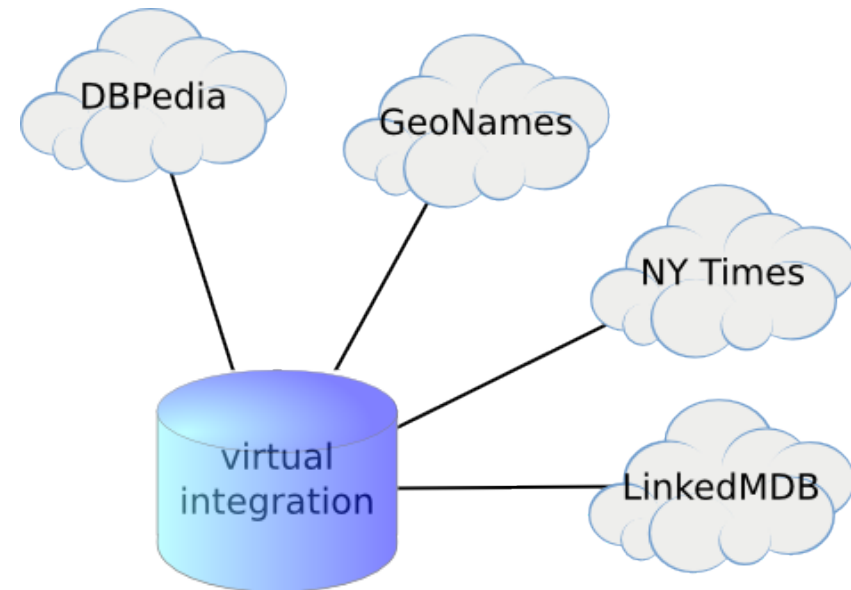


# Naive verteilte Anfragebearbeitung

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

- Benutze Ergebnisse für 3. Triple-Statement (Nested-Loop)
- $150 \times 4$  Requests  
e.g., `SELECT ?CapitalPop WHERE {ex:Berlin ex:population ?CapitalPop .}`



# Naive verteilte Anfragebearbeitung

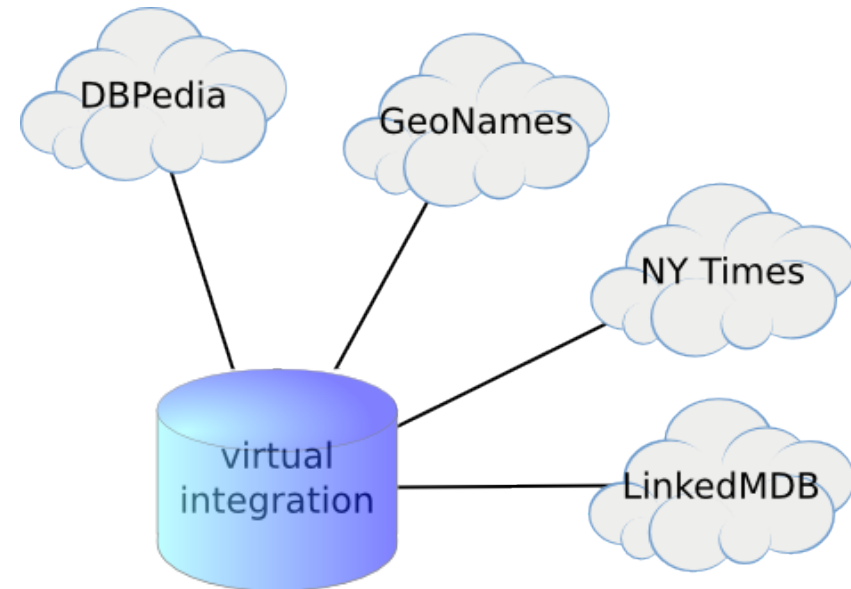
## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Insgesamt:

$$4 + 200 \times 4 + 150 \times 4 = 1404$$

Requests



# Naive verteilte Anfragebearbeitung

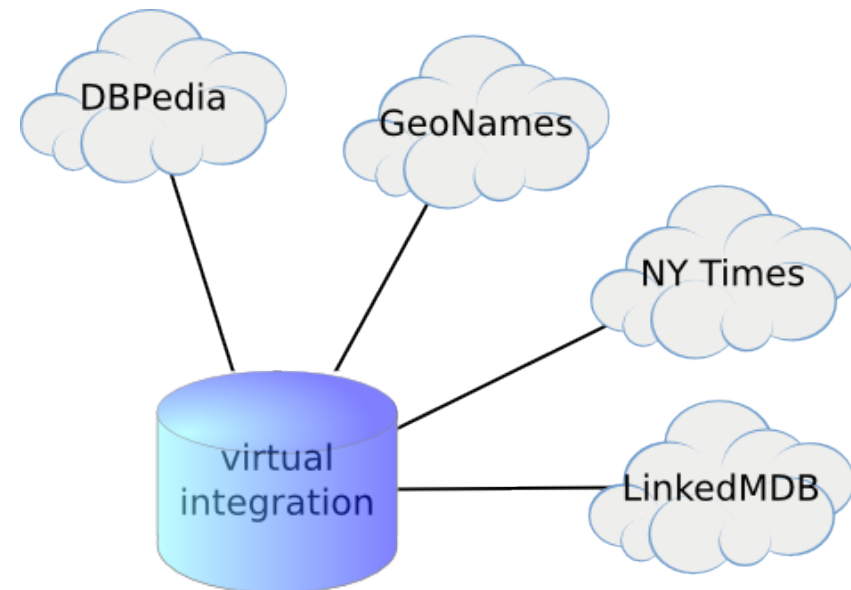
## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Insgesamt:

$$4 + 200 \times 4 + 150 \times 4 = 1404$$

Requests



Viele (unnötige) Requests an Datenquellen

# Anfrageoptimierung

## Heuristik

- Minimiere Kommunikationskosten

## Methoden

- Vereinfachung und Filter Pushing
- Source Selection
- Remote Subqueries
- Optimierung der Joinreihenfolge
- Operator-Implementierungen

# Anfrageoptimierung

## Heuristik

- Minimiere Kommunikationskosten

## Methoden

- Vereinfachung und Filter Pushing
- Source Selection
- Remote Subqueries
- Optimierung der Joinreihenfolge
- Operator-Implementierungen

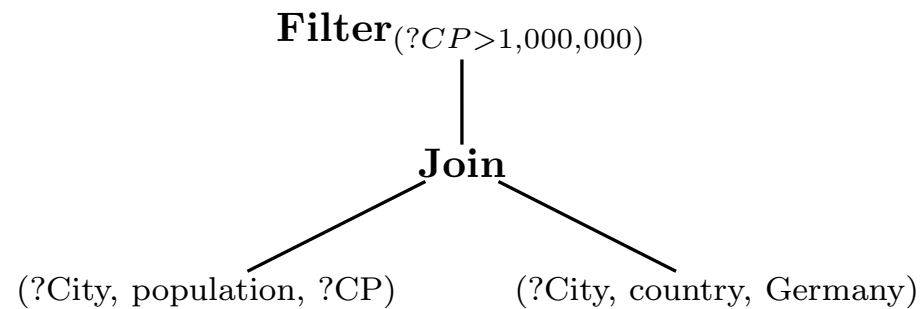
# Vereinfachung und Filter Pushing

## Normalisierung und Vereinfachung

- Widersprüchliche Prädikate
- Prädikate, die immer false/true ergeben
- Irrelevante Variablen
- etc.

# Vereinfachung und Filter Pushing

Frühe Evaluierung von Filtern reduziert die Größe der Zwischenergebnisse





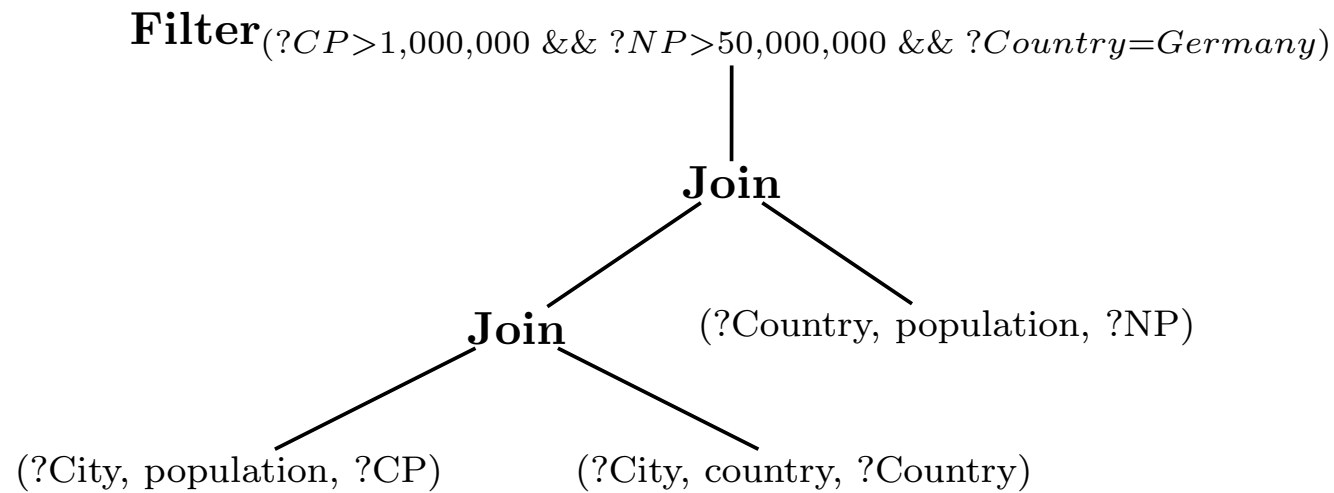
# Vereinfachung und Filter Pushing

Frühe Evaluierung von Filtern reduziert die Größe der Zwischenergebnisse



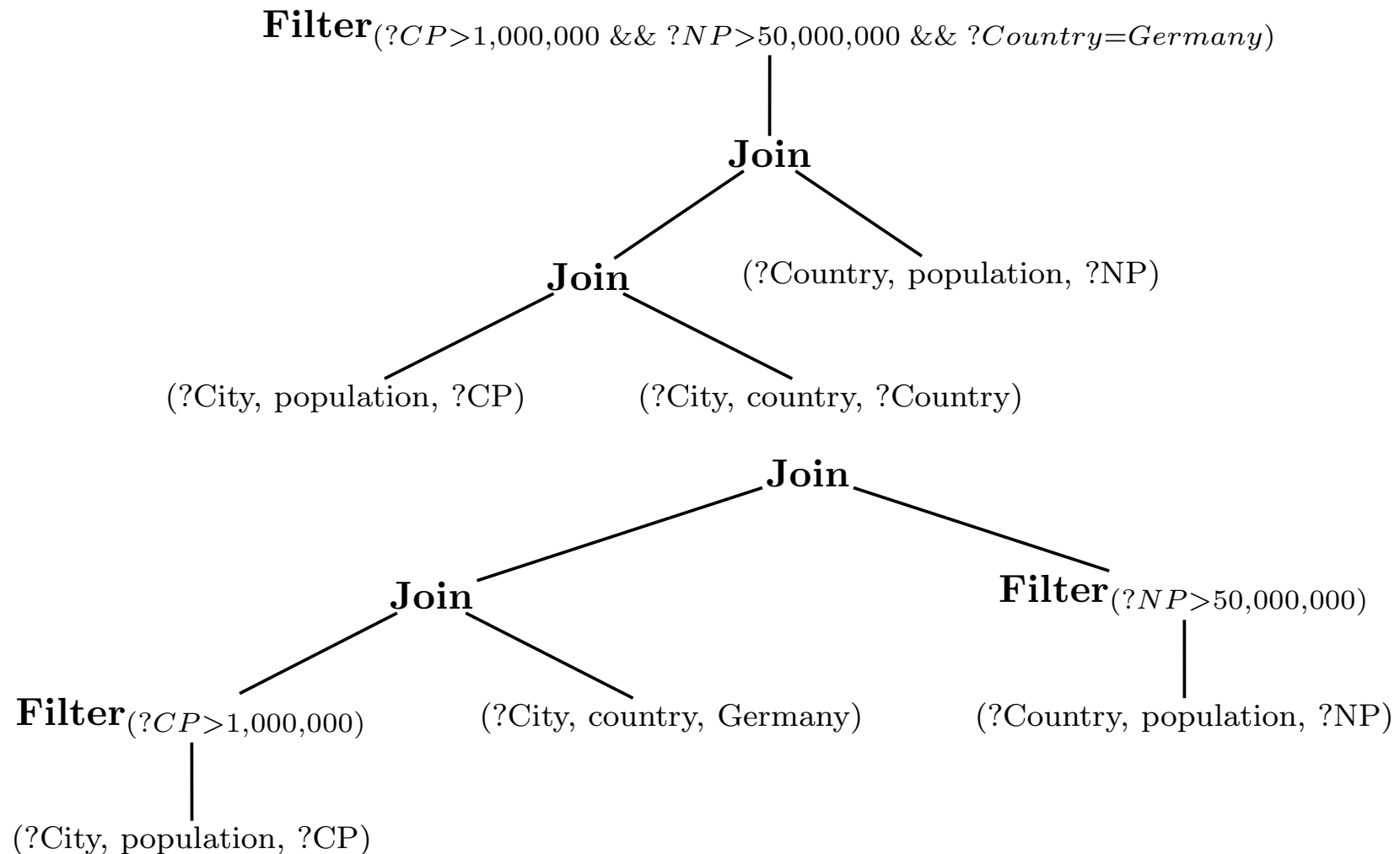
# Vereinfachung und Filter Pushing

Frühe Evaluierung von Filtern reduziert die Größe der Zwischenergebnisse



# Vereinfachung und Filter Pushing

Frühe Evaluierung von Filtern reduziert die Größe der Zwischenergebnisse



# Source Selection

## Ziel

- Identifikation von Quellen mit für die Anfrage relevanten Daten

## Ansätze

- Keine zusätzlichen Informationen
- SPARQL ASK Requests und Caching
- Statistiken und Indexe

# Source Selection

## Ziel

- Identifikation von Quellen mit für die Anfrage relevanten Daten

## Ansätze

- Keine zusätzlichen Informationen
- SPARQL ASK Requests und Caching
- Statistiken und Indexe

## Indexe und Statistiken

- Keyword-Indexes
- Schema-Indexe
- URI indexe
- Häufige Pfade
- Service-Level Descriptions
- VoID Statistics (Vocabulary of Interlinked Datasets)
- Histogramme

# Source Selection

## Ziel

- Identifikation von Quellen mit für die Anfrage relevanten Daten

## Ansätze

- Keine zusätzlichen Informationen
- SPARQL ASK Requests und Caching
- Statistiken und Indexe

## Anwendbarkeit

Es kommt sehr stark auf den Grad der Kooperation der Quellen an!

# Source Selection mit SPARQL ASK

Keine Kooperation der Quellen nötig

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Für jedes Triple-Pattern jeweils eine Anfrage an jede Quelle

Antwort: true/false

```
ASK {
  ?Country ex:capital ?Capital .
}
```

# Source Selection mit SPARQL ASK

Keine Kooperation der Quellen nötig

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Für jedes Triple-Pattern jeweils eine Anfrage an jede Quelle

Antwort: true/false

```
ASK {
  ?Country ex:capital ?Capital .
}
```

```
ASK {
  ?Country ex:population ?CountryPop .
}
```



# Source Selection mit SPARQL ASK

Keine Kooperation der Quellen nötig

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Für jedes Triple-Pattern jeweils eine Anfrage an jede Quelle

Antwort: true/false

```
ASK {
  ?Country ex:capital ?Capital .
}
```

```
ASK {
  ?Country ex:population ?CountryPop .
}
```

```
ASK {
  ?Capital ex:population ?CapitalPop .
}
```

# Source Selection mit VoID Descriptions

Beispiel DBpedia<sup>2</sup> – Kooperation der Quellen nötig

- Prefixes

@prefix owl: <http://www.w3.org/2002/07/owl#>.

@prefix rdf: <<http://www.w3.org/1999/02/22-rdf-syntax-ns#>> .

@prefix dbpedia: <<http://dbpedia.org/resource/>> .

@prefix dbpo: <<http://dbpedia.org/ontology/>> .

...

- General Information

- Basic Statistics

- Predicate Statistics

- Class Statistics

---

<sup>2</sup>[http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1\\_subset-void.n3?r=119](http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1_subset-void.n3?r=119)

# Source Selection mit VoID Descriptions

Beispiel DBpedia<sup>2</sup> – Kooperation der Quellen nötig

- Prefixes
- General Information  
    `void:sparqlEndpoint <http://dbpedia.org/sparql> ;`  
    ...
- Basic Statistics
- Predicate Statistics
- Class Statistics

---

<sup>2</sup>[http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1\\_subset-void.n3?r=119](http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1_subset-void.n3?r=119)

# Source Selection mit VoID Descriptions

Beispiel DBpedia<sup>2</sup> – Kooperation der Quellen nötig

- Prefixes
- General Information
- Basic Statistics

```
void:triples "43620475" xsd:integer ;  
void:entities "2222456" xsd:integer ;  
void:properties "1063" xsd:integer ;  
void:distinctSubjects "9495865" xsd:integer ;  
void:distinctObjects "13636604" xsd:integer ;
```

- Predicate Statistics
- Class Statistics

---

<sup>2</sup>[http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1\\_subset-void.n3?r=119](http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1_subset-void.n3?r=119)

# Source Selection mit VoID Descriptions

Beispiel DBpedia<sup>2</sup> – Kooperation der Quellen nötig

- Prefixes
- General Information
- Basic Statistics
- Predicate Statistics

```
void:propertyPartition [  
  void:property dbpo:aSide ;  
  void:triples "1600" xsd:integer ;  
  void:distinctSubjects "1552" xsd:integer ;  
  void:distinctObjects "1554" xsd:integer  
] , [  
  void:property dbpo:abbreviation ;  
  void:triples "1144" xsd:integer ;  
  void:distinctSubjects "1141" xsd:integer ;  
  void:distinctObjects "1096" xsd:integer  
] , [  
  ...  
];
```

- Class Statistics

# Source Selection mit VoID Descriptions

Beispiel DBpedia<sup>2</sup> – Kooperation der Quellen nötig

- Prefixes
- General Information
- Basic Statistics
- Predicate Statistics
- Class Statistics

```
void:classPartition [  
  void:class dbpo:Activity ;  
  void:entities "1234" xsd:integer  
] , [  
  void:class dbpo:Actor ;  
  void:entities "37898" xsd:integer  
] , [  
  ...  
]
```

---

<sup>2</sup>[http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1\\_subset-void.n3?r=119](http://code.google.com/p/fbench/source/browse/trunk/EvalBenchmark/suites/SPLendid/void/dbpedia3.5.1_subset-void.n3?r=119)

# Remote Subqueries

So viel wie möglich von den SPARQL-Endpunkten berechnen lassen und Größe von Zwischenergebnissen minimieren

- Parallele Ausführung
- Verringerte Netzwerklast
- Lastverteilung
- Bessere Antwortzeit

## Optimierungen

- Exclusive Join Groups
- Identifiziere Teilanfragen, die mehrmals im Anfragegraph enthalten sind

# Remote Subqueries

So viel wie möglich von den SPARQL-Endpunkten berechnen lassen und Größe von Zwischenergebnissen minimieren

- Parallele Ausführung
- Verringerte Netzwerklast
- Lastverteilung
- Bessere Antwortzeit

Optimierungen

- Exclusive Join Groups
- Identifiziere Teilanfragen, die mehrmals im Anfragegraph enthalten sind



# Exclusive Join Groups

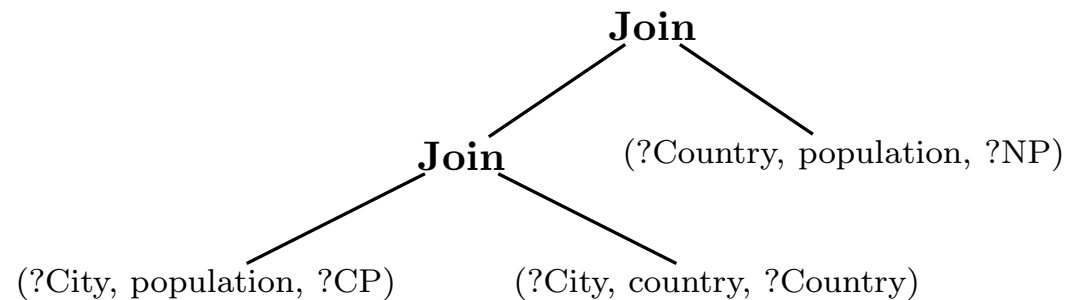
## Exclusive Join Groups

- Gruppen von Triple-Statements mit nur einer relevanten Quelle
- Kombiniere diese Triple-Statements in einer Teilanfrage

# Exclusive Join Groups

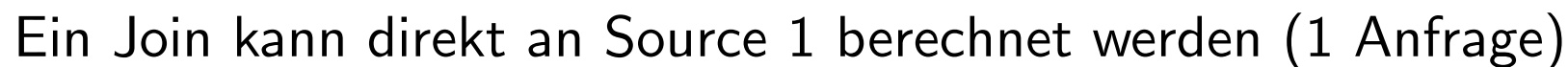
## Exclusive Join Groups

- Gruppen von Triple-Statements mit nur einer relevanten Quelle
- Kombiniere diese Triple-Statements in einer Teilanfrage



## 50 / 63

- Gruppen von Triple-Statements mit nur einer relevanten Quelle
- Kombiniere diese Triple-Statements in einer Teilanfrage



# Joinreihenfolge

## Kriterien

- Kleine Zwischenergebnisse, hohe Selektivität
- Exclusive Join Groups
- Variable Counting
- Kardinalitätsabschätzungen mit Statistiken

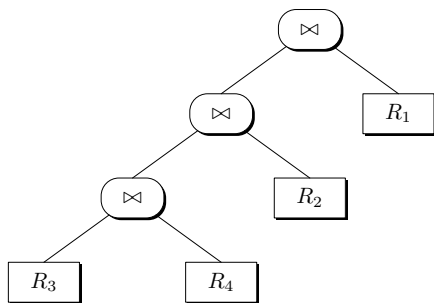
# Joinreihenfolge

## Kriterien

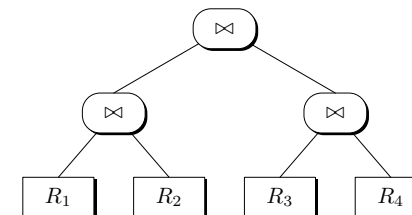
- Kleine Zwischenergebnisse, hohe Selektivität
- Exclusive Join Groups
- Variable Counting
- Kardinalitätsabschätzungen mit Statistiken

## Standardtechniken

- Lineare Bäume vs. buschige Bäume
- Kostenmodell mit Statistiken
- Dynamic Programming



Linearer Joinbaum



Buschiger Joinbaum

# Kosten- und Kardinalitätsabschätzungen

Kostenmodell für Kommunikationskosten (verteilte Datenbanken)

Beispiel:

- Source S1 relevant für Triple-Statement  $q_1$
- Source S2 relevant für Triple-Statement  $q_2$

Kosten Coordinator Join:

$$c = \text{card}(q_1) \cdot c_t + \text{card}(q_2) \cdot c_t + 2 \cdot c_m$$

Kosten Coordinator Semi-Join mit Request pro Binding:

$$c = \text{card}(q_1) \cdot c_t + \text{card}(q_1) \cdot c_m + \text{card}(q'_2) \cdot c_t$$

Kostenfaktoren:

- $c_t$ : Kosten für den Transfer eines Triples
- $c_m$ : Kosten für den Transfer einer Nachricht
- $\text{card}(q'_2)$ : Join-Partner an Source 2 für Tuple von Source 1

# Kosten- und Kardinalitätsabschätzungen

Kostenmodell für Kommunikationskosten (verteilte Datenbanken)

Beispiel:

- Source S1 relevant für Triple-Statement  $q_1$
- Source S2 relevant für Triple-Statement  $q_2$

Kosten Coordinator Join:

$$c = \text{card}(q_1) \cdot c_t + \text{card}(q_2) \cdot c_t + 2 \cdot c_m$$

Kosten Coordinator Semi-Join mit Request pro Binding:

$$c = \text{card}(q_1) \cdot c_t + \text{card}(q_1) \cdot c_m + \text{card}(q'_2) \cdot c_t$$

Kostenfaktoren:

- $c_t$ : Kosten für den Transfer eines Triples
- $c_m$ : Kosten für den Transfer einer Nachricht
- $\text{card}(q'_2)$ : Join-Partner an Source 2 für Tuple von Source 1



# Kosten- und Kardinalitätsabschätzungen

Kardinalitätsabschätzungen mit VoID-Statistiken [GS11]

Exakte Kardinalitäten sind gegeben für

- Gebundene Predicates ( $?,p,?$ )  
VoID Predicate Statistics
- Prädikat `rdf:type` mit gebundenem Object ( $?,rdf:type,o$ )  
VoID Class Statistics

Abschätzungen für alles andere

Beispiel ( $s,p,?$ )

- $card(s,p,?) = \frac{card(?,p,?)}{\text{dist. subjects for } p}$ ; beides in VoID Predicate Statistics

Beispiel ( $s,?,?$ )

- $card(s,?,?) = \frac{\text{triple in source}}{\text{dist. subjects in source}}$ ; beides in VoID General Statistics

Abschätzungen für Joins

# Kosten- und Kardinalitätsabschätzungen

Kardinalitätsabschätzungen mit VoID-Statistiken [GS11]

Exakte Kardinalitäten sind gegeben für

- Gebundene Predicates ( $?,p,?$ )  
VoID Predicate Statistics
- Prädikat `rdf:type` mit gebundenem Object ( $?,rdf:type,o$ )  
VoID Class Statistics

Abschätzungen für alles andere

Beispiel ( $s,p,?$ )

- $card(s, p, ?) = \frac{card(?, p, ?)}{\text{dist. subjects for } p}$ ; beides in VoID Predicate Statistics

Beispiel ( $s,?,?$ )

- $card(s, ?, ?) = \frac{\text{triple in source}}{\text{dist. subjects in source}}$ ; beides in VoID General Statistics

Abschätzungen für Joins

# Operator-Implementierungen

## Join-Implementierungen

- Coordinator Join
  - Daten von Quellen downloaden
  - Join lokal berechnen
- Remote Join
  - Exclusive Join Groups
- Coordinator Semi-Join
  - Evaluiere ein Triple-Pattern
  - Benutze Teilergebnisse, um weitere Triple von den Sourcen anzufragen
  - Single vs. Block Requests
  - Berechne Join am Koordinator
- Independent Join Groups
  - Erweiterung Coordinator Semi-Join
  - Reduktion der Nachrichten
  - Mehrere Joins parallel

# Operator-Implementierungen

## Join-Implementierungen

- Coordinator Join
  - Daten von Quellen downloaden
  - Join lokal berechnen
- Remote Join
  - Exclusive Join Groups
- Coordinator Semi-Join
  - Evaluiere ein Triple-Pattern
  - Benutze Teilergebnisse, um weitere Triple von den Sourcen anzufragen
  - Single vs. Block Requests
  - Berechne Join am Koordinator
- Independent Join Groups
  - Erweiterung Coordinator Semi-Join
  - Reduktion der Nachrichten
  - Mehrere Joins parallel

# Operator-Implementierungen

## Join-Implementierungen

- Coordinator Join
  - Daten von Quellen downloaden
  - Join lokal berechnen
- Remote Join
  - Exclusive Join Groups
- Coordinator Semi-Join
  - Evaluiere ein Triple-Pattern
  - Benutze Teilergebnisse, um weitere Triple von den Sourcen anzufragen
  - Single vs. Block Requests
  - Berechne Join am Koordinator
- Independent Join Groups
  - Erweiterung Coordinator Semi-Join
  - Reduktion der Nachrichten
  - Mehrere Joins parallel

# Operator-Implementierungen

## Join-Implementierungen

- Coordinator Join
  - Daten von Quellen downloaden
  - Join lokal berechnen
- Remote Join
  - Exclusive Join Groups
- Coordinator Semi-Join
  - Evaluiere ein Triple-Pattern
  - Benutze Teilergebnisse, um weitere Triple von den Sourcen anzufragen
  - Single vs. Block Requests
  - Berechne Join am Koordinator
- Independent Join Groups
  - Erweiterung Coordinator Semi-Join
  - Reduktion der Nachrichten
  - Mehrere Joins parallel

# Operator-Implementierungen

## Join-Implementierungen

- Coordinator Join
  - Daten von Quellen downloaden
  - Join lokal berechnen
- Remote Join
  - Exclusive Join Groups
- Coordinator Semi-Join
  - Evaluiere ein Triple-Pattern
  - Benutze Teilergebnisse, um weitere Triple von den Sourcen anzufragen
  - Single vs. Block Requests
  - Berechne Join am Koordinator
- Independent Join Groups
  - Erweiterung Coordinator Semi-Join
  - Reduktion der Nachrichten
  - Mehrere Joins parallel

# Coordinator Semi-Join mit Blöcken

## Coordinator Semi-Join

- Naive Variante
  - Ein Binding (Variablenwert) pro Request
- Effizientere Variante [SHH<sup>+</sup>11]
  - Mehrere Bindings pro Request (Blöcke)



# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

## Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

## Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

## Teilanfrage Block Coordinator Join:

(ex:Germany, ex:population, ?Population)

UNION

(ex:USA, ex:population, ?Population)

UNION

(ex:China, ex:population, ?Population)

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

### Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

### Teilanfrage Block Coordinator Join:

(ex:Germany, ex:population, ?Population)

UNION

(ex:USA, ex:population, ?Population)

UNION

(ex:China, ex:population, ?Population)

### Ergebnisse zu S2:

?Population=308,745,538

?Population=81,757,600

?Population=1,339,724,852

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

### Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

### Teilanfrage Block Coordinator Join:

(ex:Germany, ex:population, ?Population)

UNION

(ex:USA, ex:population, ?Population)

UNION

(ex:China, ex:population, ?Population)

### Ergebnisse zu S2:

?Population=308,745,538

?Population=81,757,600

?Population=1,339,724,852

### Problem:

- Zufällige Reihenfolge der Ergebnisse

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

### Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

### Teilanfrage Block Coordinator Join:

(ex:Germany, ex:population, ?Population)

UNION

(ex:USA, ex:population, ?Population)

UNION

(ex:China, ex:population, ?Population)

### Ergebnisse zu S2:

?Population=308,745,538

?Population=81,757,600

?Population=1,339,724,852

## Lösung:

- Variablen umbenennen

# Coordinator Semi-Join mit Blöcken

## Beispiel

S1: ?Country ex:capital ?Capital

S2: ?Country ex:population ?CountryPop

S3: ?Capital ex:population ?CapitalPop

### Ergebnisse zu S1:

?Country=ex:Germany, ?Capital=ex:Berlin

?Country=ex:USA, ?Capital=ex:WashingtonDC

?Country=ex:China, ?Capital=ex:Beijing

...

### Teilanfrage Block Coordinator Join:

(ex:Germany, ex:population, ?Population\_1)

UNION

(ex:USA, ex:population, ?Population\_2)

UNION

(ex:China, ex:population, ?Population\_3)

### Ergebnisse zu S2:

?Population\_2=308,745,538

?Population\_1=81,757,600

?Population\_3=1,339,724,852

# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```



# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

## Ergebnisse zu S1:

```
?Country=ex:Germany, ?Capital=ex:Berlin
?Country=ex:USA, ?Capital=ex:WashingtonDC
?Country=ex:China, ?Capital=ex:Beijing
...
```

# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

Ergebnisse zu S1:

```
?Country=ex:Germany, ?Capital=ex:Berlin
?Country=ex:USA, ?Capital=ex:WashingtonDC
?Country=ex:China, ?Capital=ex:Beijing
...
```

Teilanfrage zur Evaluierung von S2  
und S3:

```
(ex:Germany, ex:population, ?CountryPop_0)
UNION
(ex:Berlin, ex:population, ?CapitalPop_1)
```

# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

## Ergebnisse zu S1:

```
?Country=ex:Germany, ?Capital=ex:Berlin
?Country=ex:USA, ?Capital=ex:WashingtonDC
?Country=ex:China, ?Capital=ex:Beijing
...
```

## Teilanfrage zur Evaluierung von S2 und S3:

```
(ex:Germany, ex:population, ?CountryPop_0)
UNION
(ex:Berlin, ex:population, ?CapitalPop_1)
```

## Ergebnisse zu S2 und S3:

```
?CapitalPop_1=3,500,000
?CountryPop_0=83,000,000 ...
```

# Independent Join Groups [SHH<sup>+</sup>11]

Gruppieren *unabhängige* Triple-Statements zu einer Teilanfrage

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

## Ergebnisse zu S1:

```
?Country=ex:Germany, ?Capital=ex:Berlin
?Country=ex:USA, ?Capital=ex:WashingtonDC
?Country=ex:China, ?Capital=ex:Beijing
...
```

## Teilanfrage zur Evaluierung von S2 und S3:

```
(ex:Germany, ex:population, ?CountryPop_0)
UNION
(ex:Berlin, ex:population, ?CapitalPop_1)
```

## Ergebnisse zu S2 und S3:

```
?CapitalPop_1=3,500,000
?CountryPop_0=83,000,000 ...
```

Verbesserung: Blöcke von Bindings statt Einzelanfragen

# SPARQL-Erweiterungen

Strategien bisher

- SPARQL 1.0
- Keine Kommunikation der Quellen untereinander
- Alles läuft über Koordinator

Erweiterungen

- SPARQL 1.1 BINDINGS Operator
- SPARQL 1.1 SERVICE Operator

# SPARQL-Erweiterungen

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

## Beispielanfrage mit SERVICE Operator

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  SERVICE <http://example.org/exampleEndpoint>
    { ?Capital ex:population ?CapitalPop }
}
```

# SPARQL-Erweiterungen

## Beispielanfrage

```
SELECT ?Country ?Capital
      ?CountryPop ?CapitalPop WHERE {
  ?Country ex:capital ?Capital .
  ?Country ex:population ?CountryPop .
  ?Capital ex:population ?CapitalPop .
}
```

## Teilanfrage zur Evaluierung von S2:

```
SELECT ?Population_1 ?Population_2
      ?Population_3 WHERE {
  { ex:Germany ex:population ?Population_1 }
  UNION
  { ex:USA ex:population ?Population_2 }
  UNION
  { ex:China ex:population ?Population_3 }
}
```

## Beispielanfrage mit BINDINGS Operator

```
SELECT ?Country ?CountryPop WHERE {
  ?Country ex:population ?CountryPop .
  BINDINGS ?Country {
    (ex:Germany)
    (ex:USA)
    (ex:China)
  }
}
```



# Zusammenfassung

|                  | <b>Data<br/>Warehouses</b> | <b>Suchmaschinen</b> | <b>Lookup-basierte<br/>Anfragebearbeitung</b> | <b>Verteilte<br/>Anfragebearbeitung</b> |
|------------------|----------------------------|----------------------|---|---|
| Source Interface | hptsl. RDF-Dumps           | alle                 | Dereferenz. URIs                              | SPARQL-Endpoints                        |
| Live-Zugriff     | nein                       | nein                 | ja  | ja                                      |
| Hilfsdaten       | lok. Statistiken           | gecrawlerter Index   | –   | Index & Statistiken                     |
| Antwortzeit      | schnell                    | schnell              | medium  | medium                                  |
| Durchsatz        | schnell                    | schnell              | langsam                                       | medium                                  |
| Aktualität       | schlecht                   | medium               | hoch  | medium/hoch                             |

In Anlehnung an [HL10]

# Zusammenfassung

- Es gibt verschiedene Zugriffsarten für Linked Open Data  
*SPARQL-Endpoints, Dereferenzieren von URIs, RDF dumps*
- Es gibt verschiedene Arten auf die Daten zuzugreifen und Anfragen zu bearbeiten  
*Browsing, Suchmaschinen, unstrukturierte und strukturierte Anfragen, Lookup-basierte Verfahren, verteilte Anfragebearbeitung*
- Optimierungstechniken ähnlich zu verteilten Datenbanken  
*Kostenmodell, Kardinalitäten, Operatorimplementierungen*
- Optimierungspotential wird durch die Quellen bestimmt  
*Indexe, SPARQL 1.0 / 1.1*
- Aktualität vs. Geschwindigkeit

# Zusammenfassung

- Es gibt verschiedene Zugriffsarten für Linked Open Data  
*SPARQL-Endpoints, Dereferenzieren von URIs, RDF dumps*
- Es gibt verschiedene Arten auf die Daten zuzugreifen und Anfragen zu bearbeiten  
*Browsing, Suchmaschinen, unstrukturierte und strukturierte Anfragen, Lookup-basierte Verfahren, verteilte Anfragebearbeitung*
- Optimierungstechniken ähnlich zu verteilten Datenbanken  
*Kostenmodell, Kardinalitäten, Operatorimplementierungen*
- Optimierungspotential wird durch die Quellen bestimmt  
*Indexe, SPARQL 1.0 / 1.1*
- Aktualität vs. Geschwindigkeit

# Zusammenfassung

- Es gibt verschiedene Zugriffsarten für Linked Open Data  
*SPARQL-Endpoints, Dereferenzieren von URIs, RDF dumps*
- Es gibt verschiedene Arten auf die Daten zuzugreifen und Anfragen zu bearbeiten  
*Browsing, Suchmaschinen, unstrukturierte und strukturierte Anfragen, Lookup-basierte Verfahren, verteilte Anfragebearbeitung*
- Optimierungstechniken ähnlich zu verteilten Datenbanken  
*Kostenmodell, Kardinalitäten, Operatorimplementierungen*
- Optimierungspotential wird durch die Quellen bestimmt  
*Indexe, SPARQL 1.0 / 1.1*
- Aktualität vs. Geschwindigkeit

# Zusammenfassung

- Es gibt verschiedene Zugriffsarten für Linked Open Data  
*SPARQL-Endpoints, Dereferenzieren von URIs, RDF dumps*
- Es gibt verschiedene Arten auf die Daten zuzugreifen und Anfragen zu bearbeiten  
*Browsing, Suchmaschinen, unstrukturierte und strukturierte Anfragen, Lookup-basierte Verfahren, verteilte Anfragebearbeitung*
- Optimierungstechniken ähnlich zu verteilten Datenbanken  
*Kostenmodell, Kardinalitäten, Operatorimplementierungen*
- Optimierungspotential wird durch die Quellen bestimmt  
*Indexe, SPARQL 1.0 / 1.1*
- Aktualität vs. Geschwindigkeit

# Zusammenfassung

- Es gibt verschiedene Zugriffsarten für Linked Open Data  
*SPARQL-Endpoints, Dereferenzieren von URIs, RDF dumps*
- Es gibt verschiedene Arten auf die Daten zuzugreifen und Anfragen zu bearbeiten  
*Browsing, Suchmaschinen, unstrukturierte und strukturierte Anfragen, Lookup-basierte Verfahren, verteilte Anfragebearbeitung*
- Optimierungstechniken ähnlich zu verteilten Datenbanken  
*Kostenmodell, Kardinalitäten, Operatorimplementierungen*
- Optimierungspotential wird durch die Quellen bestimmt  
*Indexe, SPARQL 1.0 / 1.1*
- Aktualität vs. Geschwindigkeit

# Literatur I

- [DD99] Ruxandra Domenig and Klaus R. Dittrich. An overview and classification of mediated query systems. *SIGMOD Record*, 28(3):63–72, 1999.
- [GS11] Olaf Görlitz and Steffen Staab. SPLENDID: SPARQL Endpoint Federation Exploiting VOID Descriptions. In *COLD Workshop '11*, 2011.
- [Har11] Olaf Hartig. Zero-Knowledge Query Planning for an Iterator Implementation of Link Traversal Based Query Execution. In *ESWC (1)*, pages 154–169, 2011.
- [HBF09] Olaf Hartig, Christian Bizer, and Johann-Christoph Freytag. Executing SPARQL Queries over the Web of Linked Data. In *The Semantic Web - ISWC 2009*, volume 5823, pages 293–309. Springer Berlin / Heidelberg, 2009.
- [HL10] Olaf Hartig and Andreas Langeegger. A Database Perspective on Consuming Linked Data on the Web. *Datenbank-Spektrum*, 10(2):57–66, 2010.
- [LT10] Günter Ladwig and Thanh Tran. Linked Data Query Processing Strategies. In *International Semantic Web Conference (ISWC'10)*, pages 453–469, 2010.

# Literatur II

- [LT11] Günter Ladwig and Thanh Tran. SIHJoin: querying remote and local linked data. In *ESWC'11*, pages 139–153, 2011.
- [SHH<sup>+</sup>11] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. Fedx: Optimization techniques for federated query processing on linked data. In *ISWC 2011*, pages 601–616. 2011.
- [UHK<sup>+</sup>11] Jürgen Umbrich, Katja Hose, Marcel Karnstedt, Andreas Harth, and Axel Polleres. Comparing data summaries for processing live queries over linked data. *World Wide Web*, 14:495–544, 2011. [10.1007/s11280-010-0107-z](https://doi.org/10.1007/s11280-010-0107-z).