Graphbasierte Modellierung und Repräsentation eines Forschungsdateninventars

Daniel Jettka
INEL-Projekt (Indigenous Northern Eurasian Languages)
Akademie der Wissenschaften in Hamburg

Grammatiken, Korpora und Sprachtechnologie für indigene nordeurasische Sprachen

Leitung:	Prof. Dr. Beáta Wagner-Nagy
Antragssteller:	Prof. Dr. Beáta Wagner-Nagy, Dr. Michael Rießler, Hanna Hedeland, Timm Lehmberg
Wissenschaftliche Mitarbeiter:	Dr. Alexandre Arkhipov (Forschungskoordinator), Timm Lehmberg (Technischer Koordinator), Dr. Maria Brykina, Chris Lasse Däbritz, Anne Ferger, Daniel Jettka, Tiina Klooster, Svetlana Orlova

Erschließung sprachlicher Ressourcen indigener Sprachen sowie ihre Bereitstellung über eine digitale Forschungsinfrastruktur

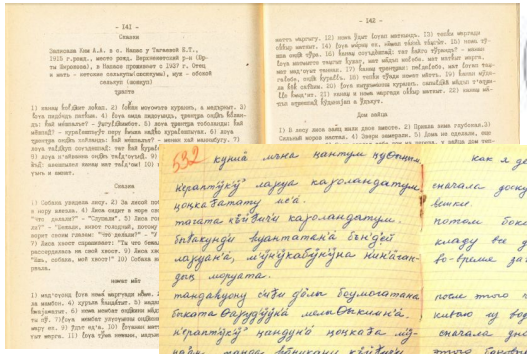
Daten

Große Mengen von Sprachdaten

- in Archiven (nicht alles öffentlich)
- in Publikationen

Daten/Texte mit und ohne Audio

Archive in Tomsk, St. Petersburg, Moskau, Hamburg, Tartu, …



Inventur

Ziel: Inventar der vorhandenen Daten

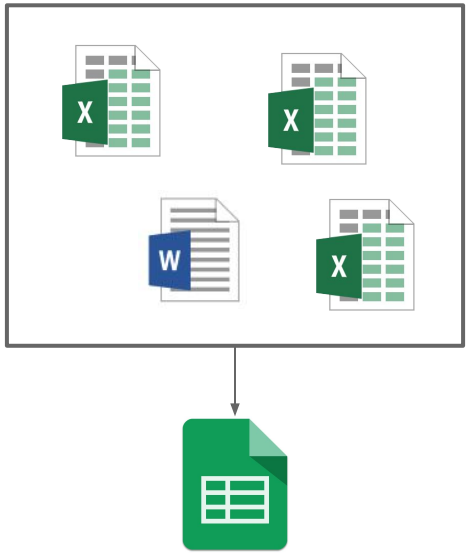
erweiterbar

durchsuchbar

web-basiert

Problem:

Mehrere fortlaufend ergänzte Inventurlisten (versch. Struktur und Formate)



Lösung:

Inventur und Zusammenführung von Inventurlisten

(Google Spreadsheet für Übergangszeit)

Inventur

Vorteile Google Spreadsheets

1. Einfache Kollaboration
2. Höheres Maß an Konsistenz durch:
   a. Auswahllisten (auf Basis vorhandener Werte/Listen)
   b. Validierung (z.B. Dopplungen)
   c. Dynamisch generierte Inhalte (z.B. volle Namen, IDs)
   d. Vglw. einfache Erweiterung der Oberfläche/Funktionalität

➔ Erste Schritte der Datenmodellierung (Entitäten, Kategorien, Referenzen)

Inventur

AKADEMIE DER WISSENSCHAFTEN IN HAMBURG

Datenmodellierung:

- Identifikation/Definition von Entitäten
- Identifikation/Definition relevanter Eigenschaften der Entitäten
- Identifikation/Definition von Relationen zw. Den Entitiäten
- Validierung/Kuration existierender Beschreibungen

Relationen zwischen Entitäten:

Item	isPartOf	Archive
Item	isPartOf	Item
Item	isCollectedBy	Person
Item	hasLanguage	Language

Person	isSpeakerOf	Item
Person	isSpeakerOf	Language
Language	isPartOf	Language

Inventur

Nachteile Google Spreadsheets

1. Beschränkungen des tabellarischen Datenmodells
2. Beschränkungen für Generierung dynamischer Inhalte
3. Datenschutz

Lösungsansatz

- Versch. Datenbanktechnologien möglich
- Evaluation der Graphdatenbank Neo4j

Graphdatenbank

- Community Edition frei verwendbar
- Einfache Installation
- Schema-los; simple Datentypen
- REST Endpoint
- Webschnittstelle für Suche und kommando-basierte Pflege von Daten

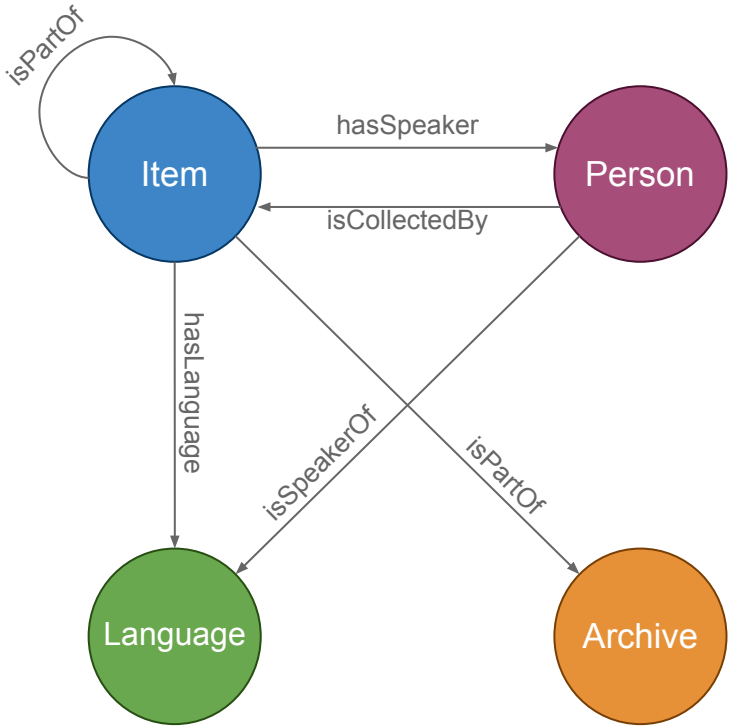
➔ Transformation existierender Beschreibungen in Cypher-Ausdrücke (JSON)
➔ Import in Neo4j-Datenbank

Graphdatenbank

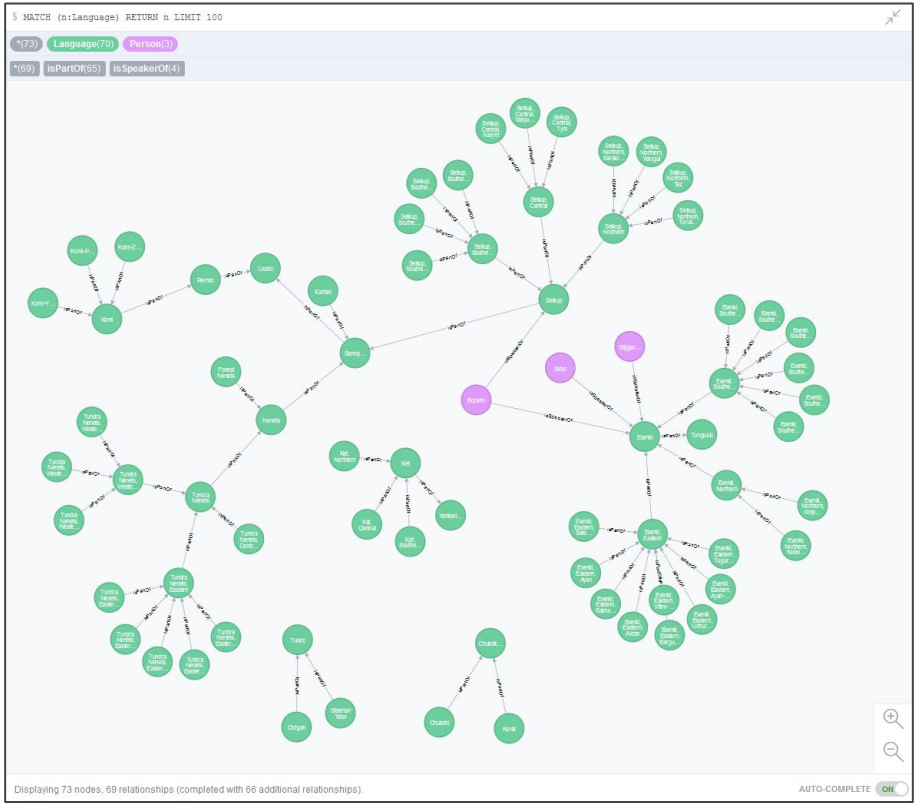
neo4j

Vorteile:

- Flexible Datenmodellierung, Repräsentation, Abfrage

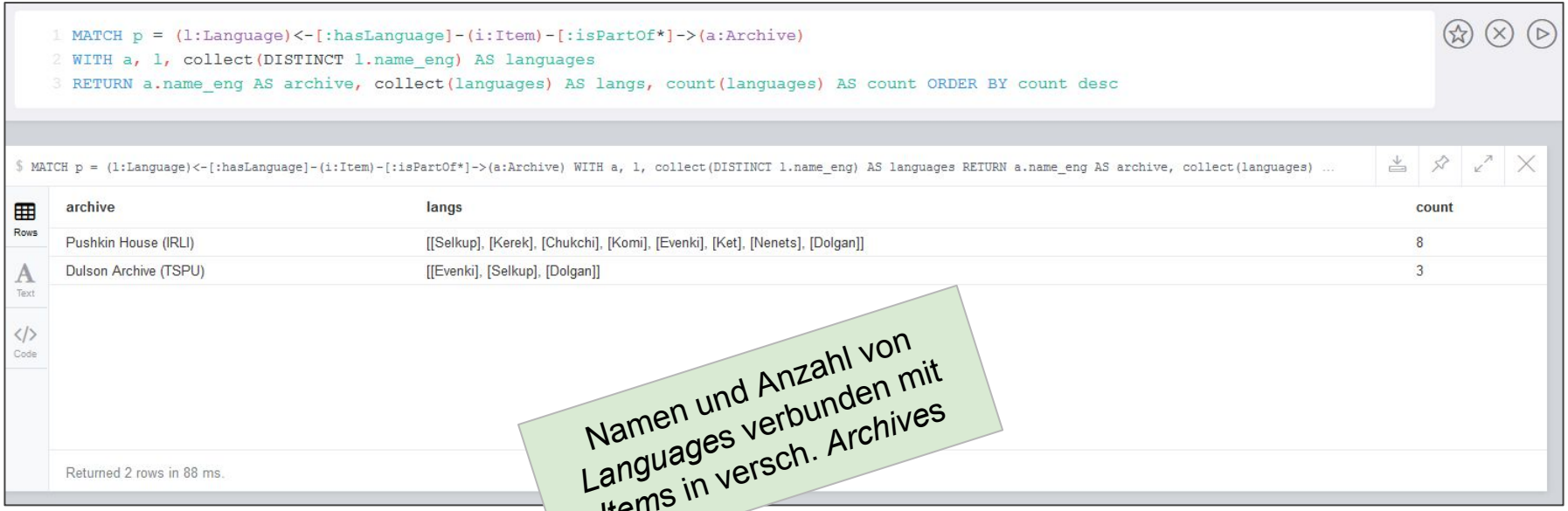


Sprachen


$ MATCH (n:Language) RETURN n LIMIT 100
*(73)
Language(70)
Person(3)
*(69)
isPartOf(65)
isSpeakerOf(4)
Displaying 73 nodes, 69 relationships (completed with 66 additional relationships).
AUTO-COMPLETE
ON

Sprachen

```
1 MATCH p = (l:Language)<-[:hasLanguage]-(i:Item)-[:isPartOf*]->(a:Archive)
2 WITH a, l, collect(DISTINCT l.name_eng) AS languages
3 RETURN a.name_eng AS archive, collect(languages) AS langs, count(languages) AS count ORDER BY count desc
```



$ MATCH p = (l:Language)<-[:hasLanguage]-(i:Item)-[:isPartOf*]->(a:Archive) WITH a, l, collect(DISTINCT l.name_eng) AS languages RETURN a.name_eng AS archive, collect(languages) ...

Rows / Text / Code

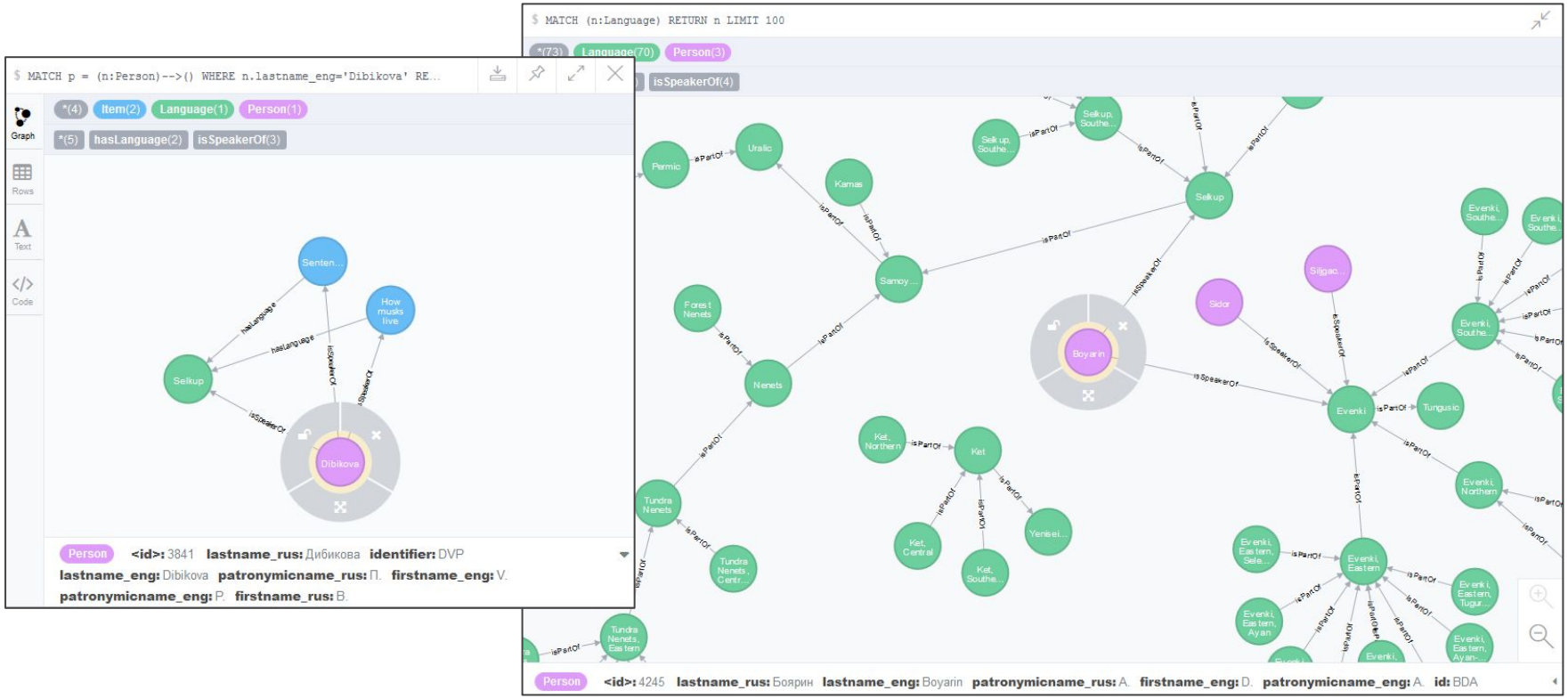
archive	langs	count
Pushkin House (IRLI)	[[Selkup], [Kerek], [Chukchi], [Komi], [Evenki], [Ket], [Nenets], [Dolgan]]	8
Dulson Archive (TSPU)	[[Evenki], [Selkup], [Dolgan]]	3

Returned 2 rows in 88 ms.

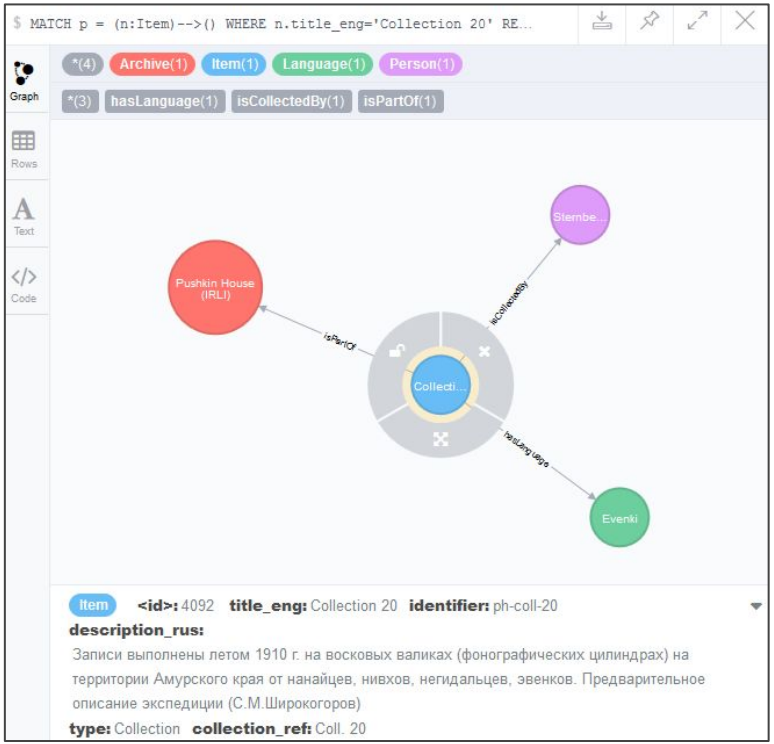
Namen und Anzahl von Languages verbunden mit Items in versch. Archives

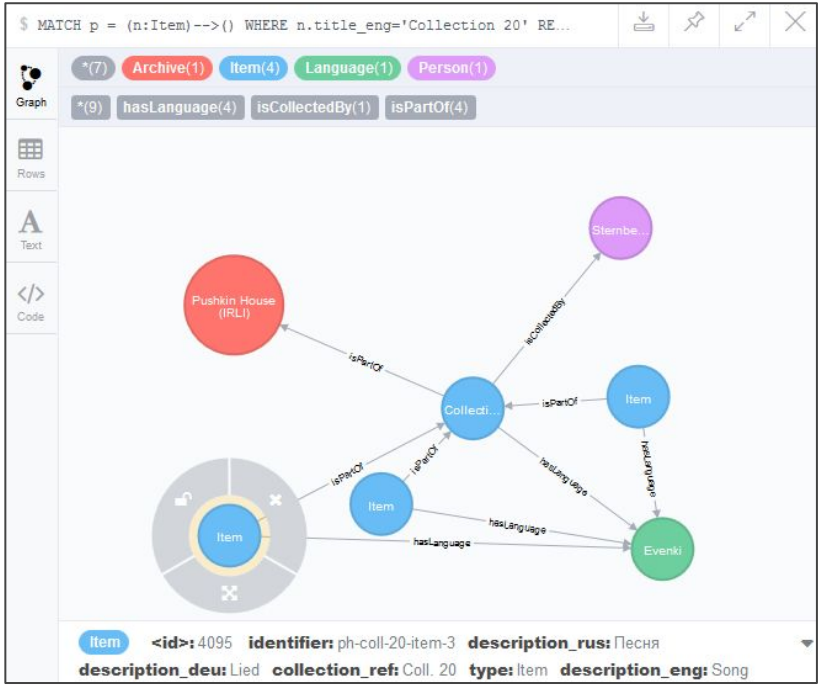
```
MATCH p = (l:Language)<-[:hasLanguage]-(i:Item)-[:isPartOf*]->(a:Archive)
WITH a, l, collect(DISTINCT l.name_eng) AS languages
RETURN a.name_eng AS archive, collect(languages) AS langs, count(languages) AS count ORDER BY count desc
```

Sprecher


$ MATCH p = (n:Person)-->() WHERE n.lastname_eng='Dibikova' RE...
*(4)
Item(2)
Language(1)
Person(1)
*(5)
hasLanguage(2)
isSpeakerOf(3)
Graph
Rows
Text
Code
Senten...
How musks live
Selkup
Dibikova
Person <id>: 3841 lastname_rus: Дибикова identifier: DVP
lastname_eng: Dibikova patronymicname_rus: П. firstname_eng: V.
patronymicname_eng: P. firstname_rus: B.
$ MATCH (n:Language) RETURN n LIMIT 100
*(73)
Language(70)
Person(3)
isSpeakerOf(4)
Permic
Uralic
Kamas
Samoy...
Selkup, Southe...
Selkup
Forest Nenets
Nenets
Boyarin
Sidor
Siljgac...
Evenki
Evenki, Southe...
Tungusic
Ket, Northern
Ket
Ket, Central
Ket, Southe...
Yenisei...
Tundra Nenets
Tundra Nenets, Centr...
Tundra Nenets, Eastern
Evenki, Northern
Evenki, Eastern, Sele...
Evenki, Eastern
Evenki, Eastern, Tugur...
Evenki, Eastern, Ayan
Evenki, Eastern, Ayan-...
isPartOf
isSpeakerOf
Person <id>: 4245 lastname_rus: Боярин lastname_eng: Boyarin patronymicname_rus: A. firstname_eng: D. patronymicname_eng: A. id: BDA

Collections





Collections

```
MATCH (l:Language)<-[:hasLanguage]-(i:Item)-[:isCollectedBy]->(p:Person)
RETURN p.firstname_eng+" "+p.patronymicname_eng+" "+p.lastname_eng AS researcher, collect(DISTINCT l.name_eng) AS langs, count(i) AS item_count
ORDER BY item_count desc
```

researcher	langs	item_count
Andrej Petrovich Duljzon	[Evenki, Selkup]	253
N. P. Maksimova	[Selkup]	126
Z. P. Demjyanenko	[Dolgan, Selkup]	67
V. V. Bykonya	[Selkup]	64
E. S. Kuznecova	[Selkup]	62
N. M. Voevodina	[Selkup]	58
I. A. Iljyashenko	[Selkup]	56
E. G. Bekker	[Selkup]	53
N. V. Denning	[Sel	32
A. A. Kim		31
G. K. Verner		30
Yu. A. Moreva		29
N. P. Beljtyukova		28
T. M. Kosheverova		19
O. A. Osipova		17
L. A. Alitkina	Selkup]	13

Returned 55 rows in 253 ms.

Anzahl von *Items* und versch. verbundenen *Languages*, die von jedem Forscher erhoben wurden

MATCH (l:Language)<-[:hasLanguage]-(i:Item)-[:isCollectedBy]->(p:Person)
RETURN p.firstname_eng+" "+p.patronymicname_eng+" "+p.lastname_eng, collect(DISTINCT l.name_eng) AS langs, count(i) AS item_count
ORDER BY item_count desc

Archive

```
MATCH p = (i:Item)-[:isPartOf*]->(a:Archive)
RETURN a.name_eng AS archive, collect(i.identifier) AS items, count(p) AS count
ORDER BY count asc
```

$ MATCH p = (i:Item)-[:isPartOf*]->(a:Archive) RETURN a.name_eng AS archive, collect(i.identifier) AS items, count(p) AS count ORDER BY count asc

Rows | Text | Code

archive	items	count
Pushkin House (IRLI)	[ph-coll-514, ph-coll-515, ph-coll-516, ph-coll-517, ph-coll-519, ph-coll-559, ph-coll-565, ph-coll-574, ph-coll-675, ph-coll-732, ph-coll-738, ph-coll-739, ph-coll-746, ph-coll-796, ph-coll-849, ph-coll-74, ph-coll-65, ph-coll-94, ph-coll-112, ph-coll-126, ph-coll-146, ph-coll-133, ph-coll-206, ph-coll-268, ph-coll-436, ph-coll-449, ph-coll-489, ph-coll-491, ph-coll-492, ph-coll-26, ph-coll-26-item-1, ph-coll-26-item-2, ph-coll-26-item-3, ph-coll-26-item-4, ph-coll-33, ph-coll-33-item-1, ph-coll-35, ph-coll-35-item-2, ph-coll-35-item-1, ph-coll-35-item-6, ph-coll-35-item-5, ph-coll-35-item-4, ph-coll-35-item-3, ph-coll-35-item-10, ph-coll-35-item-9, ph-coll-35-item-8, ph-coll-35-item-7, ph-coll-35-item-14, ph-coll-35-item-13, ph-coll-35-item-12, ph-coll-35-item-11, ph-coll-35-item-17, ph-coll-35-item-18, ph-coll-35-item-15, ph-coll-35-item-16, ph-coll-35-item-21, ph-coll-35-item-22, ph-coll-35-item-19, ph-coll-35-item-20, ph-coll-35-item-25, ph-coll-35-item-26, ph-coll-35-item-23, ph-coll-35-item-24, ph-coll-35-item-29, ph-coll-35-item-30, ph-coll-35-item-27, ph-coll-35-item-28, ph-coll-27, ph-coll-31, ph-coll-20, ph-coll-20-item-1, ph-coll-20-item-2, ph-coll-20-item-3, ph-coll-21, ph-coll-21-item-11, ph-coll-21-item-10, ph-coll-21-item-13, ph-coll-21-item-12, ph-coll-21-item-7, ph-coll-21-item-6, ph-coll-21-item-9, ph-coll-21-item-8, ph-coll-21-item-19, ph-coll-21-item-18, ph-coll-21-item-21, ph-coll-21-item-20, ph-coll-21-item-14, ph-coll-21-item-17, ph-coll-21-item-16, ph-coll-21-item-22, ph-coll-21-item-23, ph-coll-21-item-24, ph-coll-21-item-25, ph-coll-21-item-4, ph-coll-21-item-5, ph-coll-21-item-1]	99
Dulson Archive (TSPU)	[da-dolgan, da-dolgan-vol-2, da-dolgan-vol-2-item-2, da-dolgan-vol-2-item-1, da-dolgan-vol-2-item-4, da-dolgan-vol-2-item-7, da-dolgan-vol-1, da-dolgan-vol-1-item-3, da-dolgan-vol-1-item-2, da-dolgan-vol-1-item-5, da-dolgan-vol-1-item-6, da-dolgan-vol-1-item-9, da-dolgan-vol-1-item-11, da-dolgan-vol-1-item-10, da-dolgan-vol-1-item-13, da-dolgan-vol-1-item-15, da-dolgan-vol-1-item-1, da-dolgan-vol-5, da-dolgan-vol-5-item-4, da-dolgan-vol-5-item-3, da-dolgan-vol-5-item-2, da-dolgan-vol-6, da-dolgan-vol-6-item-1, da-dolgan-vol-6-item-5, da-dolgan-vol-6-item-4, da-dolgan-vol-6-item-3, da-dolgan-vol-6-item-2, da-dolgan-vol-6-item-24, da-dolgan-vol-6-item-25, da-dolgan-vol-6-item-10, da-dolgan-vol-6-item-11, da-dolgan-vol-6-item-12, da-dolgan-vol-6-item-13, da-dolgan-vol-6-item-14, da-dolgan-vol-6-item-8, da-dolgan-vol-6-item-9, da-dolgan-vol-6-item-19, da-dolgan-vol-6-item-20, da-dolgan-vol-6-item-21, da-dolgan-vol-6-item-22, da-dolgan-vol-6-item-15, da-dolgan-vol-6-item-17, da-dolgan-vol-6-item-18, da-dolgan-vol-3, da-dolgan-vol-3-item-2, da-dolgan-vol-3-item-1, da-dolgan-vol-4, da-dolgan-vol-4-item-13, da-dolgan-vol-4-item-12, da-dolgan-vol-4-item-11, da-dolgan-vol-4-item-10, da-dolgan-vol-4-item-3, da-dolgan-vol-4-item-4, da-dolgan-vol-4-item-1, da-dolgan-vol-4-item-2, da-dolgan-vol-4-item-7,	913

Returned 2 rows in 385 ms.

Anzahl und Namen der Items pro Archive

```
MATCH p = (i:Item)-[:isPartOf*]->(a:Archive)
RETURN a.name_eng AS archive, collect(i.identifier) AS item, count(p) AS count
ORDER BY count asc
```

Graphdatenbank

Nachteil:

- Beschränkte Zugänglichkeit für Linguisten (Nicht-Techniker)

Lösung:

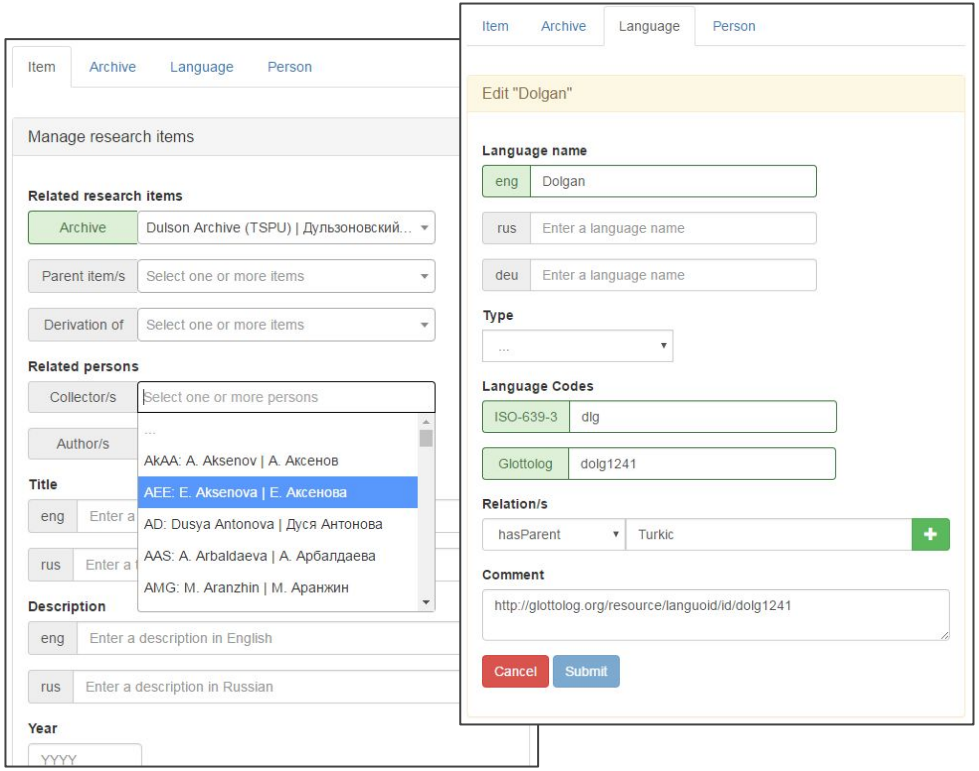
- Evaluation vorhandener GUIs (z.B. structr, Linkurious, popoto.js)
  → entweder zu teuer oder nur teilweise einsetzbar
- Implementierung einer eigenen GUI

GUI Prototyp I

Prototyp I: HTML-Formulare

- Formulare mit Bootstrap (HTML5, CSS, JavaScript)
- Validierung, Auto-completion, Felder mit Mehrfachauswahl
- Verbindung mit Neo4j REST Endpoint

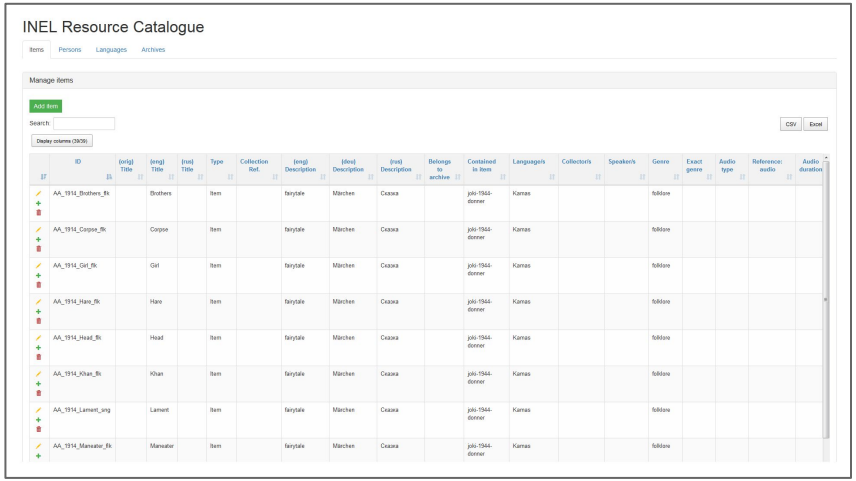
➔ Nicht nutzerfreundlich (Tabellenfunktionen gewünscht)



GUI Prototyp II

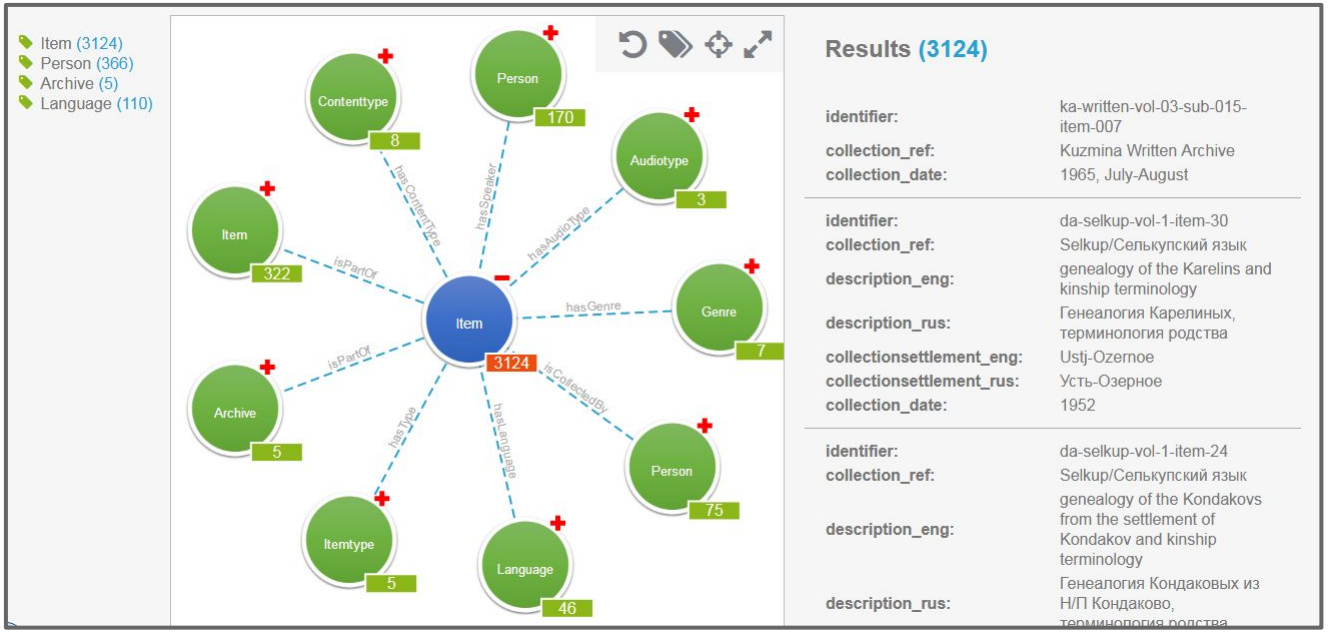
Prototyp II: neo4j + DataTables

- Einlesen von DB-Inhalten in DataTables-basierte HTML-Tabelle
- Anpassung der Tabellendaten auf Client-Seite
- Schreiben aktualisierter Daten in DB durch JavaScript-Trigger
- Zugriffsschutz via Shibboleth



Graphische Recherche in Daten

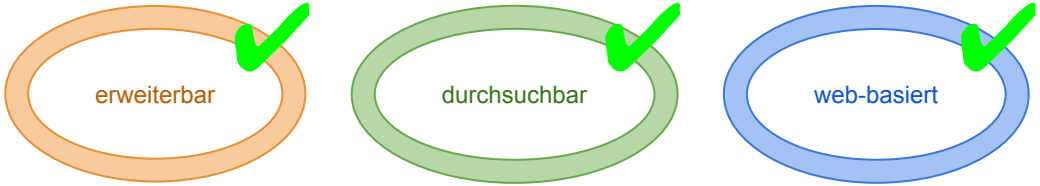
popoto.js



JavaScript-Bibliothek für Neo4j-Visualisierung

Zusammenfassung

- Inventar von Forschungsdaten in Graphdatenbank
- Prototyp einer Webschnittstelle



- Ansatz anwendbar auf andere Daten

Ausblick

- Evaluation Geo-Plugin für Neo4j
- Evaluation von Triple Stores (Graphdatenbanken für RDF)
  - z.B. Fuseki, Blazegraph
- (zusätzl.) Verwendung existierender Ontologien
- Webschnittstelle mit anderen Data Stores verbinden
- Vernetzung von Datenbeständen