Text2Scene

Object Dictionaries

Patrick Schrottenbacher

Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

Assignment

Consider how to implement the structure of the WordNet Dictionary into Neo4j.

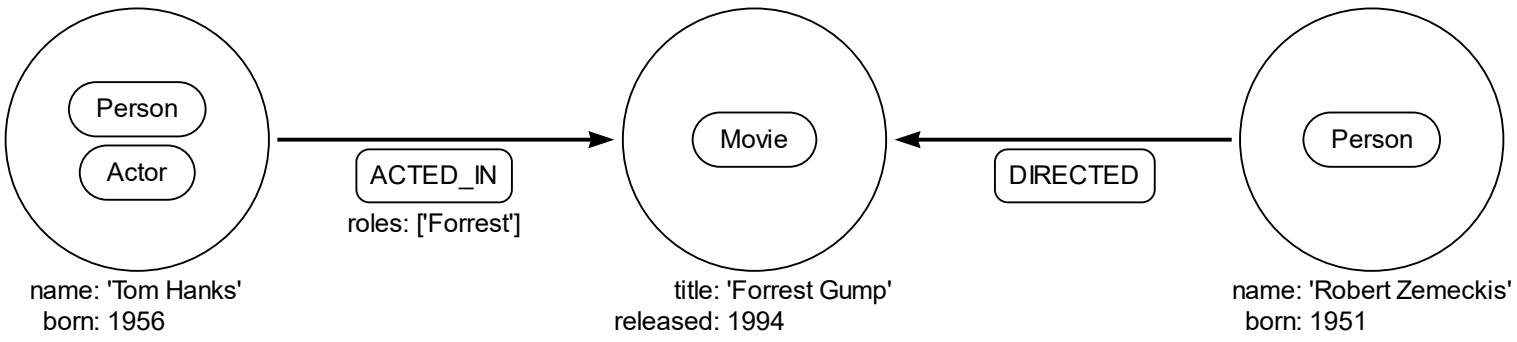
To extract words from the WordNet Dictionary there exists the python package NLTK

Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

Neo4j / Graphdatabases

- Nodes and edges
- Specifically, for neo4j: Nodes as well as edges have any number of attributes and
- Pros
  - Fast querying for (strongly) connected datasets
  - Great for modelling complex/irregular datasets
- Cons
  - Takes up more storage space (when compared to most RDBs)
  - Data structure isn‘t static



Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

NLTK/Wordnet

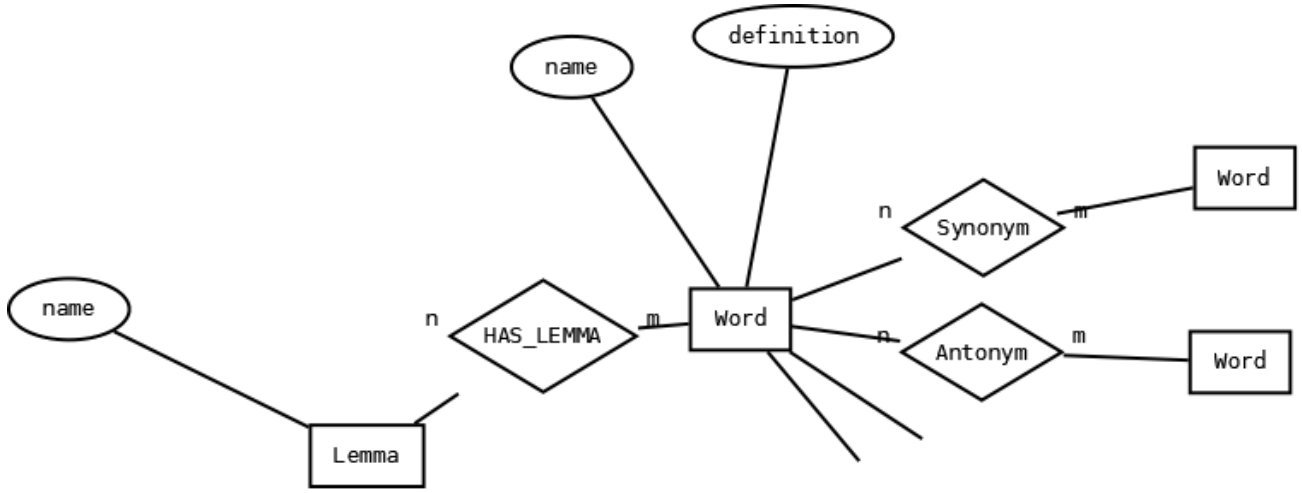
- NLTK: interface for lexical resources
- Wordnet: lexical database
- Mainly specifying relations between words via synonyms, antonyms, hyponyms etc.

Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

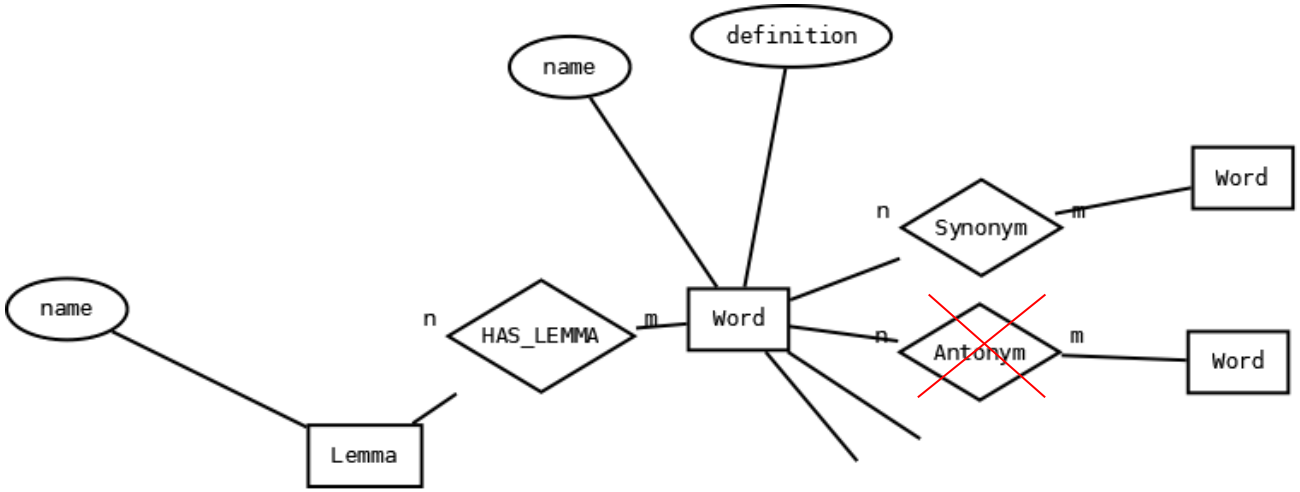
Procedure

- Packages: nltk, neomodel (neo4j)



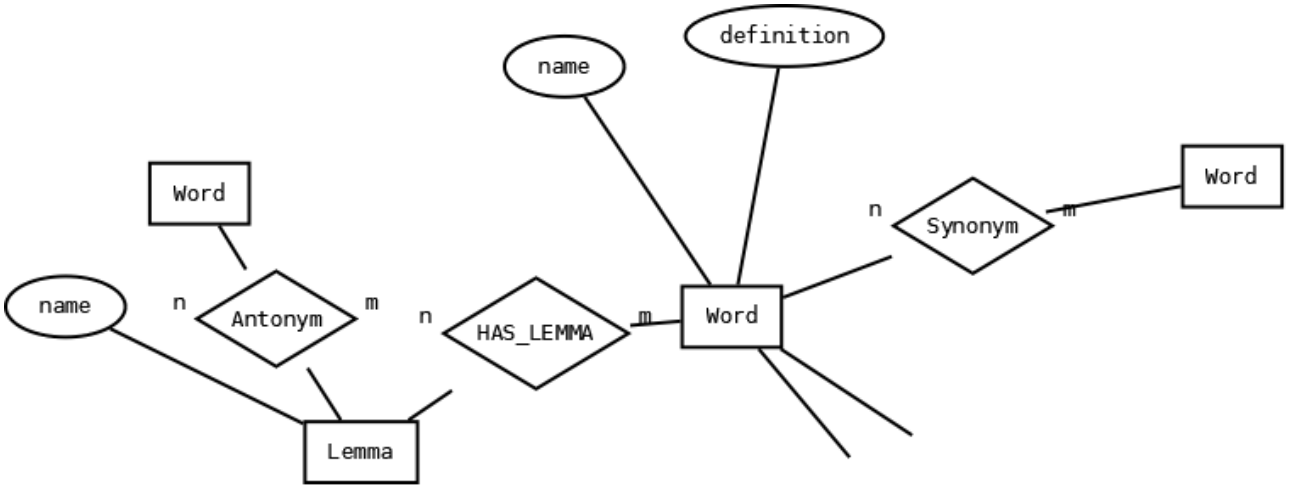
Procedure

- Packages: nltk, neomodel (neo4j)



Procedure

- Packages: nltk, neomodel (neo4j)



Procedure

Idea 1

Iterate over all synsets(words) and save them into the database, recursively add all relationships to other nodes as well as the nodes themselves.

Problems:

- Slow, since there's I/O isn't being optimized
- Possibility of very deep recursions

Procedure

Idea 2

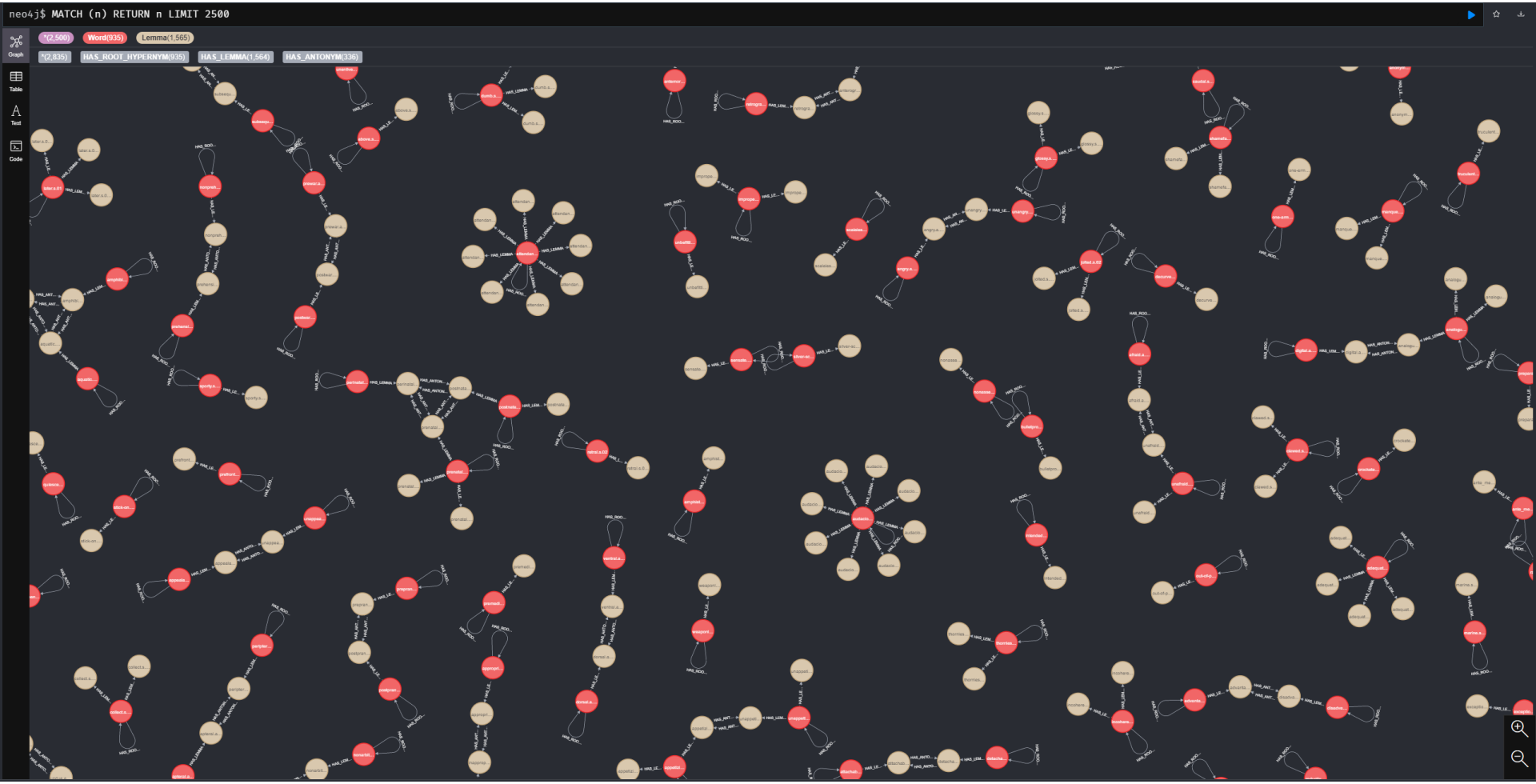
Save all synsets and their lemmas into the database. Later add all the relationships

- One can easily introduce batching to reduce I/O
- No deep recursions
- Easier to update the existing dataset

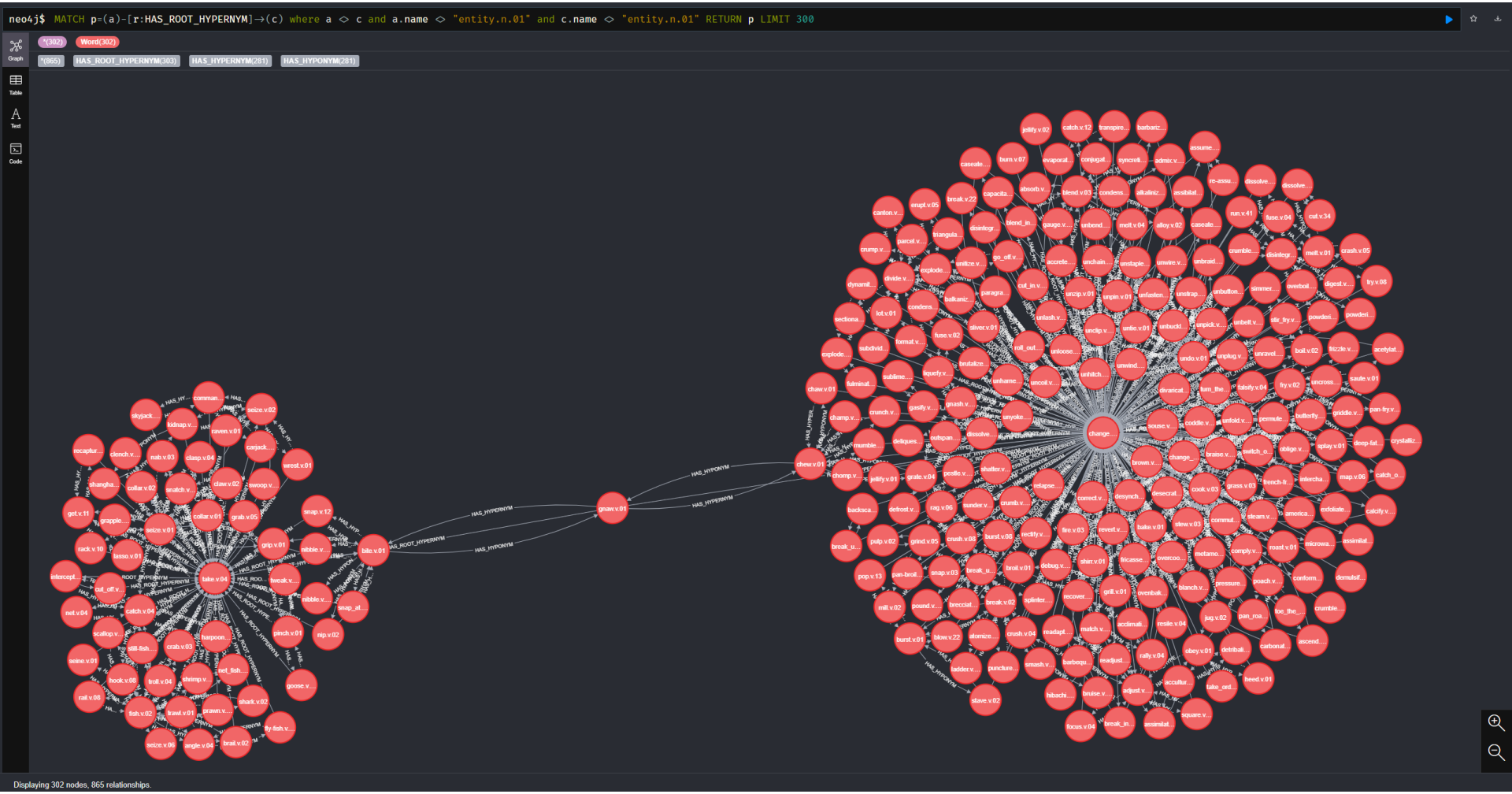
Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

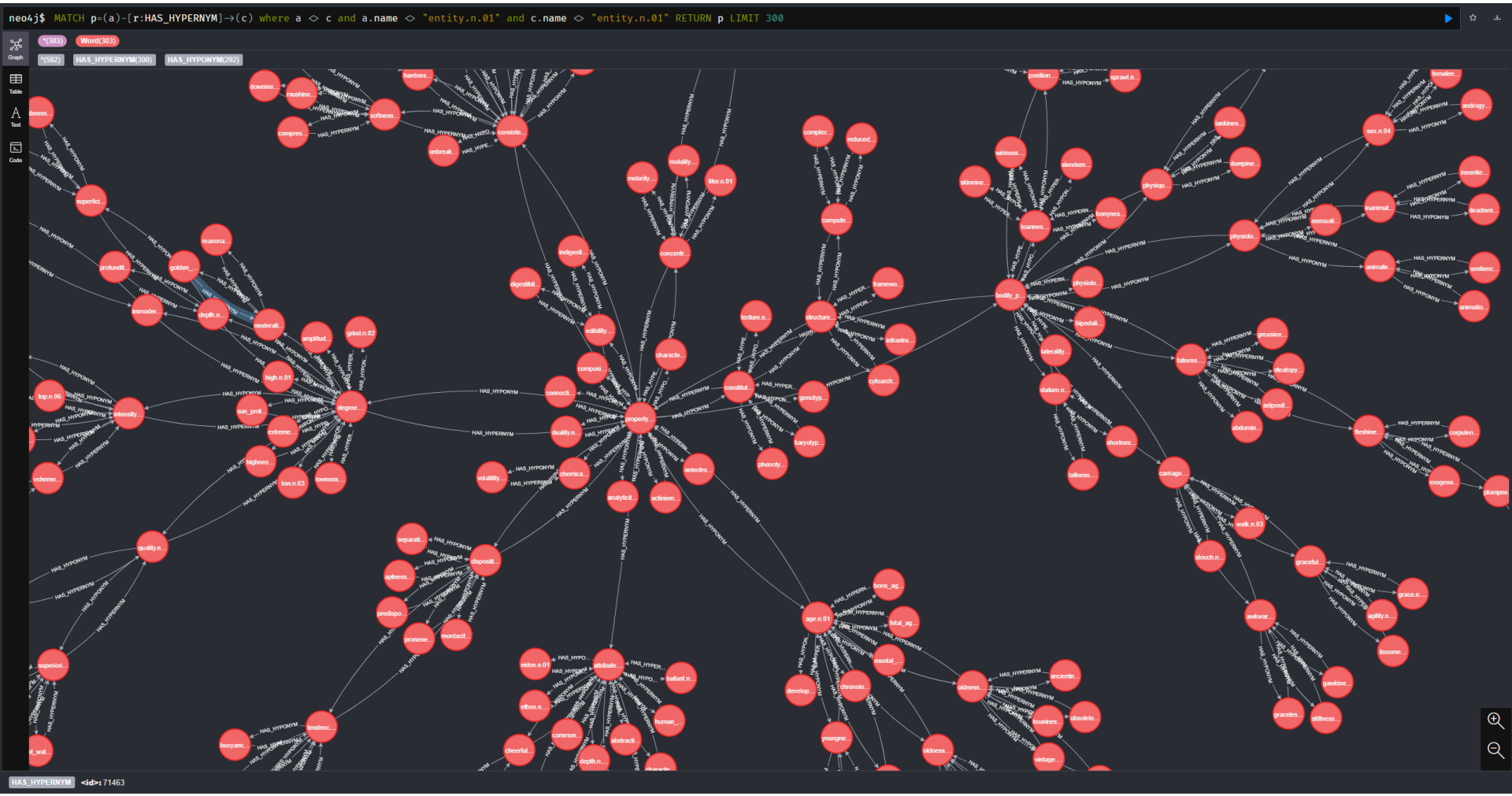
Results


neo4j$ MATCH (n) RETURN n LIMIT 2500
*(2,500)
Word(935)
Lemma(1,565)
*(2,835)
HAS_ROOT_HYPERNYM(935)
HAS_LEMMA(1,564)
HAS_ANTONYM(336)
Graph
Table
Text
Code

Results


neo4j$ MATCH p=(a)-[r:HAS_ROOT_HYPERNYM]→(c) where a <> c and a.name <> "entity.n.01" and c.name <> "entity.n.01" RETURN p LIMIT 300
*(302)
Word(302)
*(865)
HAS_ROOT_HYPERNYM(303)
HAS_HYPERNYM(281)
HAS_HYPONYM(281)
Graph
Table
Text
Code
bake.v.04
bake.v.01
gnaw.v.01
chew.v.01
change...
HAS_HYPERNYM
HAS_HYPONYM
HAS_ROOT_HYPERNYM
Displaying 302 nodes, 865 relationships.

Results


neo4j$ MATCH p=(a)-[r:HAS_HYPERNYM]→(c) where a <> c and a.name <> "entity.n.01" and c.name <> "entity.n.01" RETURN p LIMIT 300
*(303)
Word(303)
*(592)
HAS_HYPERNYM(300)
HAS_HYPONYM(292)
Graph
Table
Text
Code
HAS_HYPERNYM
<id>: 71463

Agenda

1. Assignment
2. Neo4j
3. NLTK/Wordnet
4. Procedure
5. Results
6. Sources

Sources

- neomodel: https://neomodel.readthedocs.io/_/downloads/en/stable/pdf/
- neo4j: https://neo4j.com/
- NLTK/wordnet: https://www.nltk.org/howto/wordnet.html