Capstone
Developing Knowledge Graphs from Research Abstracts

Michael Cohan

Maria Golovco

Harsha Mahankali

Jason Winkler

Agenda

- **Business Problem**
- **Considerations and Pre-analysis**
- **Data & Tools**
- **Solution-Our Knowledge Graph**
- **Insights and Results**

Business Problem

To help explore Neo4j's capability as a **Knowledge Graph** using PubMed **journal abstracts, u**in order to perform **exploratory queries** into ***'NASH'*.** Concurrently, we wanted to **understand the depth** of our **Knowledge Graph** by performing **graph algorithms** to recommend approaches to uncover **hidden information**.

Business Problem

Quick facts about NASH:

- NASH is **heavily influenced** by lifestyle.
- Studies have suggested that 80-90% of the **obese population** is at risk for developing NASH.
- **NASH** is closely related to the triple threat epidemic:
  - Pre-diabetes
  - Diabetes II
  - Obesity

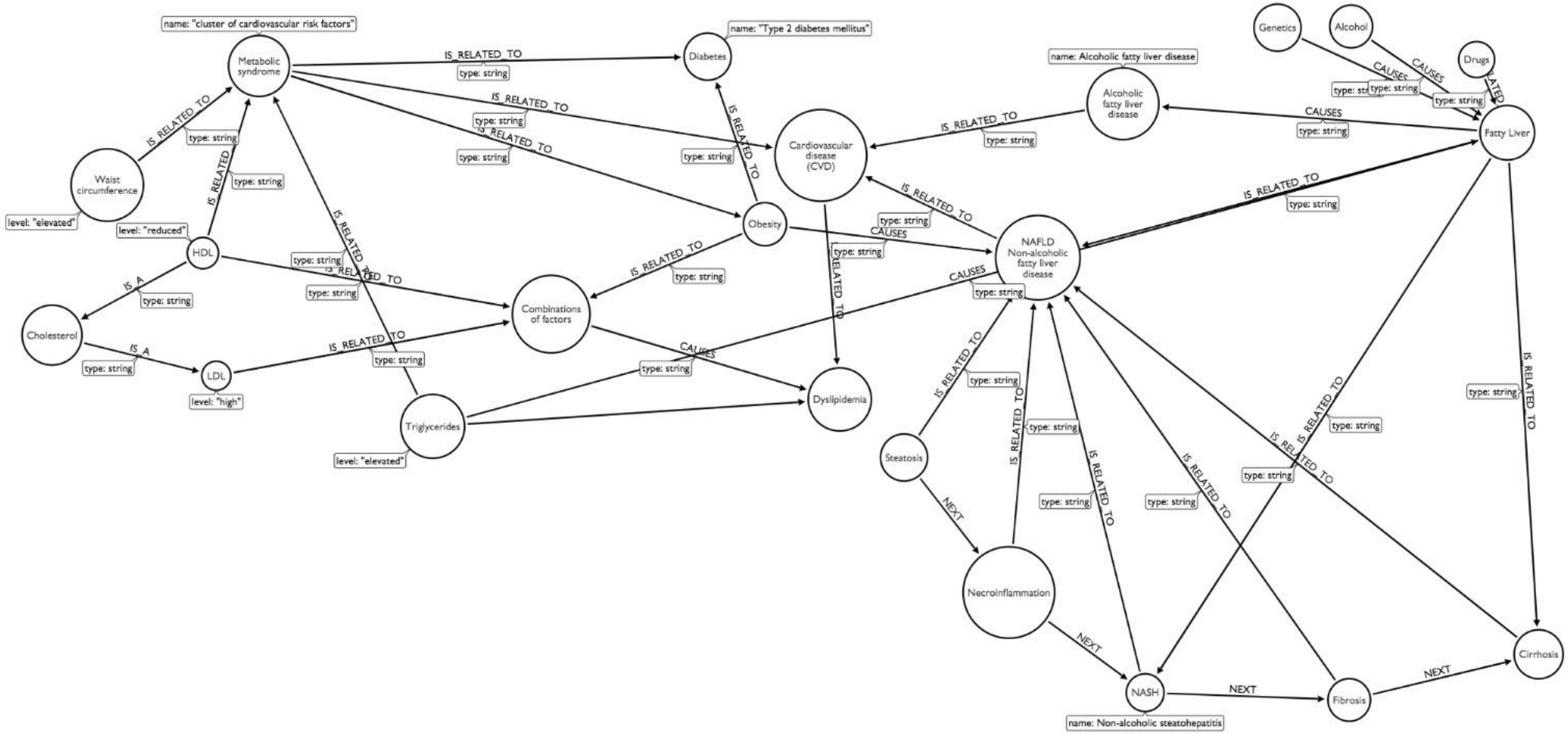
https://www.the-nash-education-program.com/what-is-nash/

Business Problem

What can gain?

- NASH's **market** is reportedly worth **$35 billion (2018).**
- The **FDA** is allowing for a **fast-track designation** for NASH related drugs.
  - Drug or treatment needs to show promise.
- **FDA & EMA** can designate **Orphan Drug Status** for rare diseases associated with NASH.
  - Government assistance if the drug treats a disease that may not be profitable.

Business Problem-Our External Knowledge Graph Design



Data & Tools

DATA:

- 1146 abstracts from Pubmed
- Datasets generated using [redacted]

1) 116,632 keys with location in each abstract
2) 40 unique ontology definitions and associated code name

-Linked together using a shared code name

TOOLS:

- Neo4j Community Edition (open-source version).
- Neo4j plugins GraphAware, APOC, and GraphAlgorithms
- Semantic Enrichers ConceptNet5 and Microsoft Concept Graphs

Ideas for Direction

- Information from nodes and relationships can be captured in, or can be transferred to, existing nodes and relationships through labels and properties
- How to deal with lost information and how to recover if possible-location
- The keys (entities) added are not contained in the texts of the abstracts and can distort true relationships if not accounted for properly. Keys that are not also named entities, tags, or Keywords
- Grouping and Relabeling nodes
- Simplify Knowledge Graph
- Importing new data-batches, procedures, users
- Building and using a Knowledge Graph is an iterative process

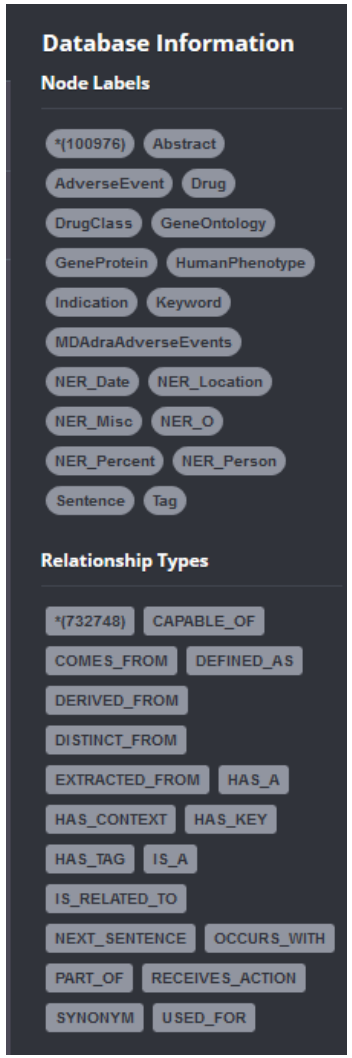
Data In Our Model

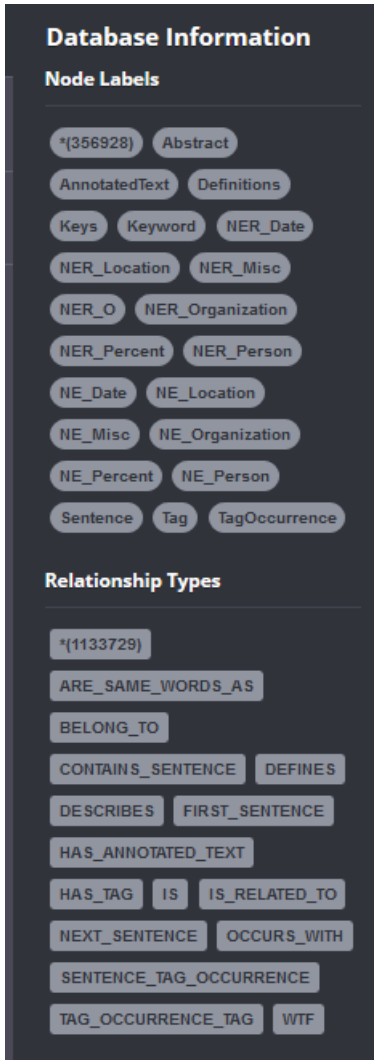
- 10,347 individual unique definition/key pairs after deduplication in Python
- All data cased to lower

class	value
adverseEvent	hepatocellular injury
adverseEvent	neuroendocrine cancer
geneProtein	chemokine receptors, ccr2 and ccr5
geneProtein	nos2a
indication	neuroendocrine cancer
mdaDraAdverseEvents	hepatocellular injury
mdaDraAdverseEvents	neuroendocrine cancer

- Focus on 8 definitions:
  - [redacted]
  - [redacted]
  - [redacted]
  - [redacted]
  - [redacted]
  - [redacted]
  - [redacted]
  - [redacted]

Solution-Refactoring

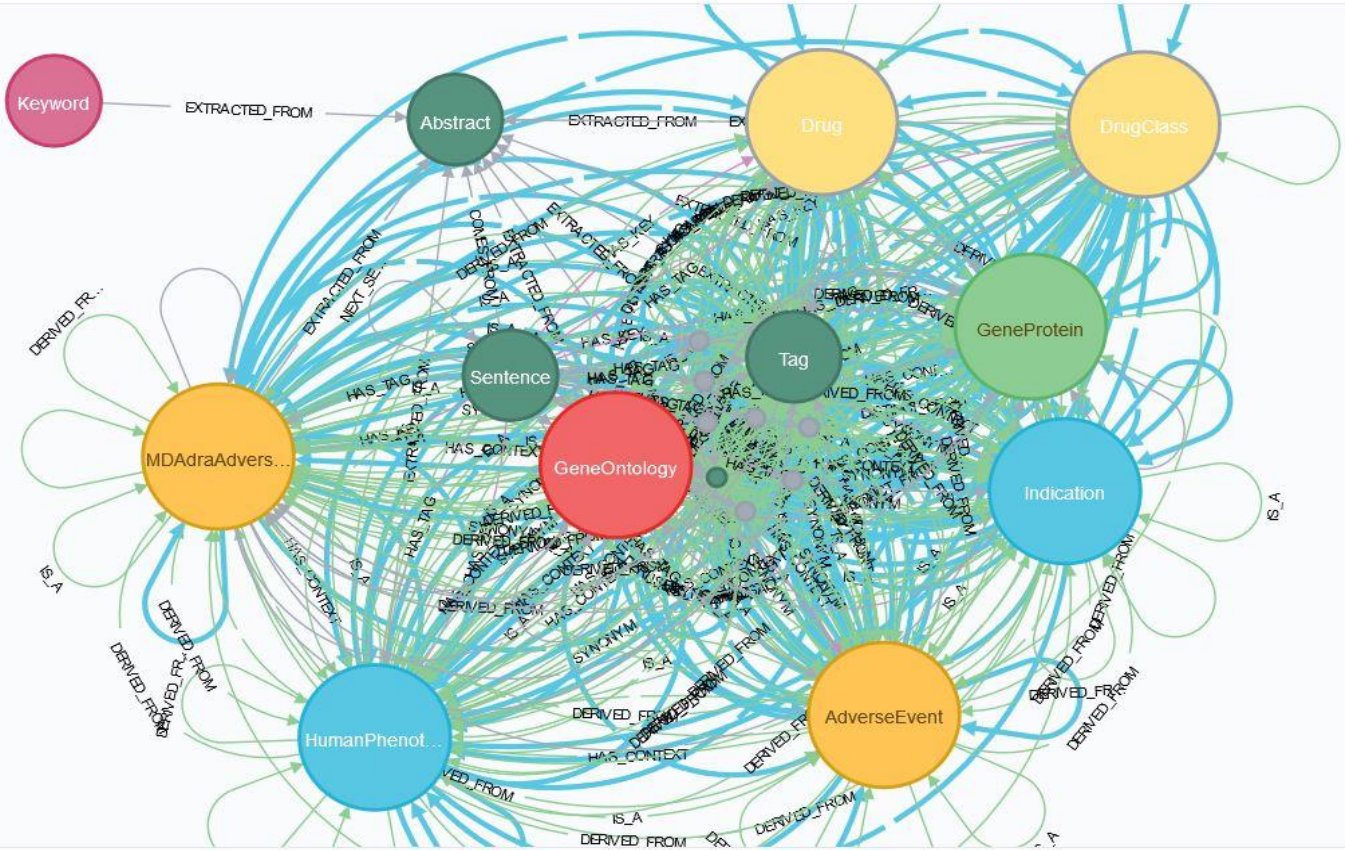




Our Knowledge Graph Schema

*(20) NER_O(1) NER_Date(1) Drug(1) NER_Location(1) Keyword(1) Indication(1) Abstract(1) NER_Misc(1) MDAdraAdverseEvents(1) DrugClass(1) NashNafld(1) HumanPhenotype(1)
NER_Person(1) AdverseEvent(1) NER_Percent(1) Tag(1) NashOccur(1) Sentence(1) GeneOntology(1) GeneProtein(1)

*(663) RECEIVES_ACTION(2) OCCURS_WITH(256) DISTINCT_FROM(6) COMES_FROM(1) EXTRACTED_FROM(10) HAS_A(4) PART_OF(4) IS_A(49) SYNONYM(18) HAS_CONTEXT(24) CAPABLE_OF(2)
HAS_KEY(9) USED_FOR(4) IS_RELATED_TO(187) DERIVED_FROM(42) DEFINED_AS(2) HAS_TAG(17) NEXT_SENTENCE(1) SIMILAR(25)



Data profiling - Metadata

- The NashNafld label was created to group nodes for all string forms of NASH and NAFLD
- Below are other labels these nodes share and the properties belonging to the nodes

Labels	Properties
["MDAdraAdverseEvents", "AdverseEvent", "Indication", "NashNafld"]	[["value", "class"], ["class", "value"]]
["Indication", "NashNafld"]	[["value", "class"]]
["Tag", "NER_O", "Indication", "NashOccur", "NashNafld"]	[["community", "freq", "partition", "ne", "pos", "lastTxId", "id", "value", "language"]]
["Tag", "NER_O", "MDAdraAdverseEvents", "AdverseEvent", "Indication", "NashNafld"]	[["community", "freq", "partition", "ne", "pos", "lastTxId", "id", "value", "language"]]
["Keyword", "Indication", "NashNafld"]	[["relevance", "keywordsList", "originalTagId", "originalTagValue", "id", "value", "numTerms"]]
["Keyword", "MDAdraAdverseEvents", "AdverseEvent", "Indication", "NashNafld"]	[["relevance", "keywordsList", "originalTagId", "originalTagValue", "id", "value", "numTerms"]]

Data profiling – graph density

Data profiling on a graph database like Neo4j gives us more insightful understandings of the actual data we are working on. The results obtained can then be used for further detailed analysis, performance tuning, database schema optimization and data migration.

$$\text{Graph Density} = \frac{Edges/}{\frac{of\ nodes*(of\ nodes-1)}{2}}$$

Global Graph Density (considering the graph one-directional)		
by calculating individual parts		
MATCH (n) RETURN count(n) AS NumberOfNodes;	nodes	400,252
MATCH ()-[r]-() RETURN count(r) as numberOfRelations;	relations	1,764,370
		3,528,740
		160,201,263,252
	nodes / relations	0.002%
by using a combined line of code		
MATCH (n1)-[r]-(n2)	distinct nodes	283,001
RETURN count(DISTINCT n1) as nrNodes,	distinct relations	882,185
count(DISTINCT r) as nrEdges,		
count(DISTINCT r)/((count(DISTINCT n1)) * (count(DISTINCT n2) - 1.0)) AS graphDensity		1,764,370
		80,089,283,000
	nodes /distinct rela	0.002%

*Figures taken from a previous iteration of our Graph Database

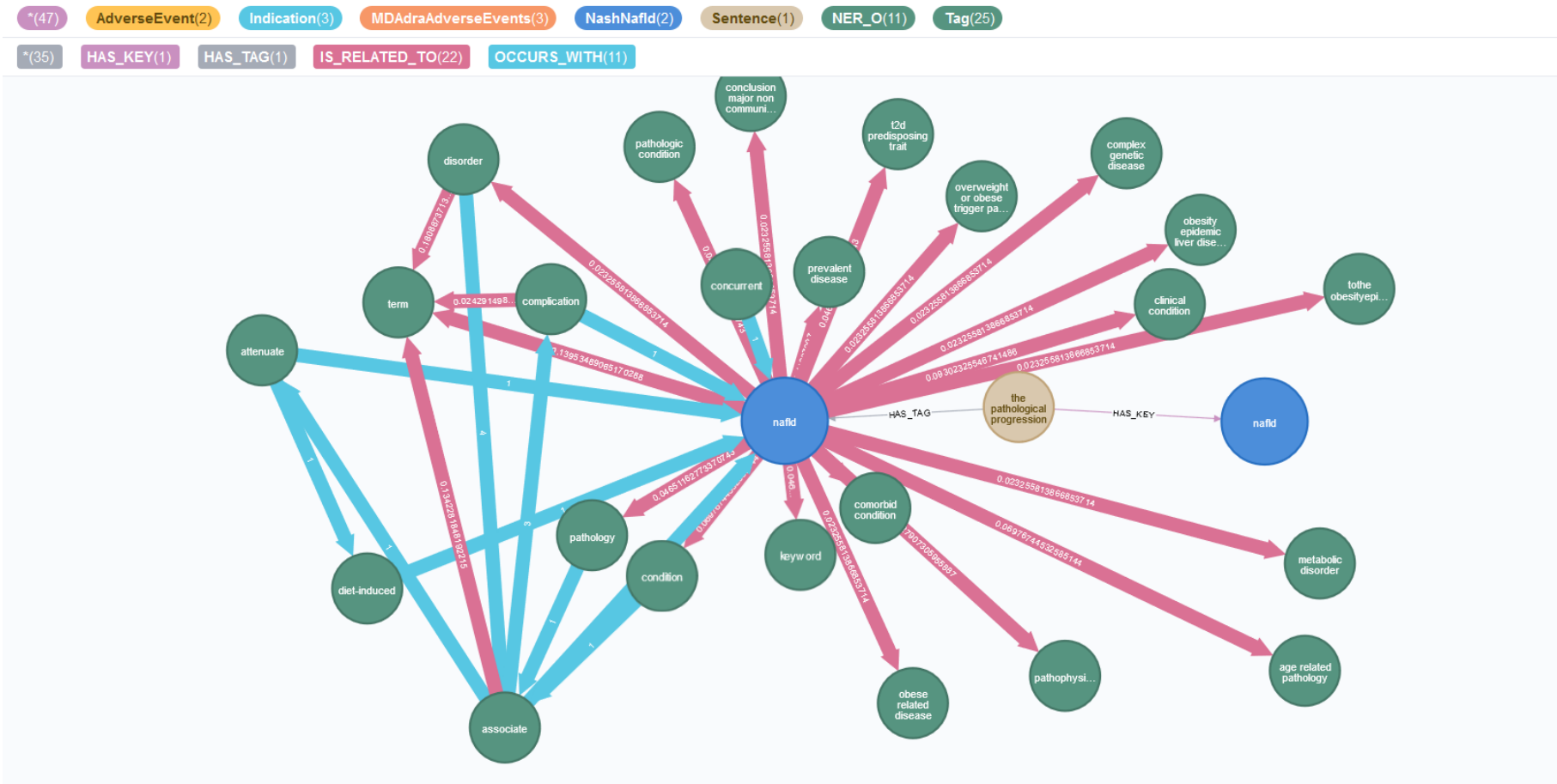
What kind of nodes exist?

- Get a random sample of data
- Sample size set to <= 10% of nodes from Label(s)

Labels	SampleSize	Avg_PropertyCount	Avg_RelationshipCount
["Tag"]	4935	7.001823708206683	1.5333333333333379
["Tag", "NER_O"]	1699	8.643319599764581	33.229546792230764
["Sentence"]	1201	4.0	38.26644462947543
["Keyword"]	467	7.0	3.7601713062098496
["GeneProtein"]	368	2.0	20.75543478260871
["Tag", "NER_O", "GeneProtein"]	222	9.0	17.5
["MDAdraAdverseEvents", "AdverseEvent", "HumanPhenotype", "Indication"]	127	2.0	41.59842519685041
["GeneOntology"]	123	3.0	28.98373983739835
["Keyword", "GeneProtein"]	123	7.0	3.3414634146341458
["Abstract"]	109	4.0	33.61467889908256
["MDAdraAdverseEvents", "AdverseEvent", "Indication"]	84	2.0	30.48809523809524
["MDAdraAdverseEvents"]	49	2.0	19.57142857142857
["MDAdraAdverseEvents", "AdverseEvent"]	47	2.0	177.7021276595745
["Drug"]	36	2.0	5.722222222222223
["Indication"]	30	2.0	31.500000000000018
["Tag", "NER_O", "GeneOntology"]	29	9.931034482758621	109.41379310344828

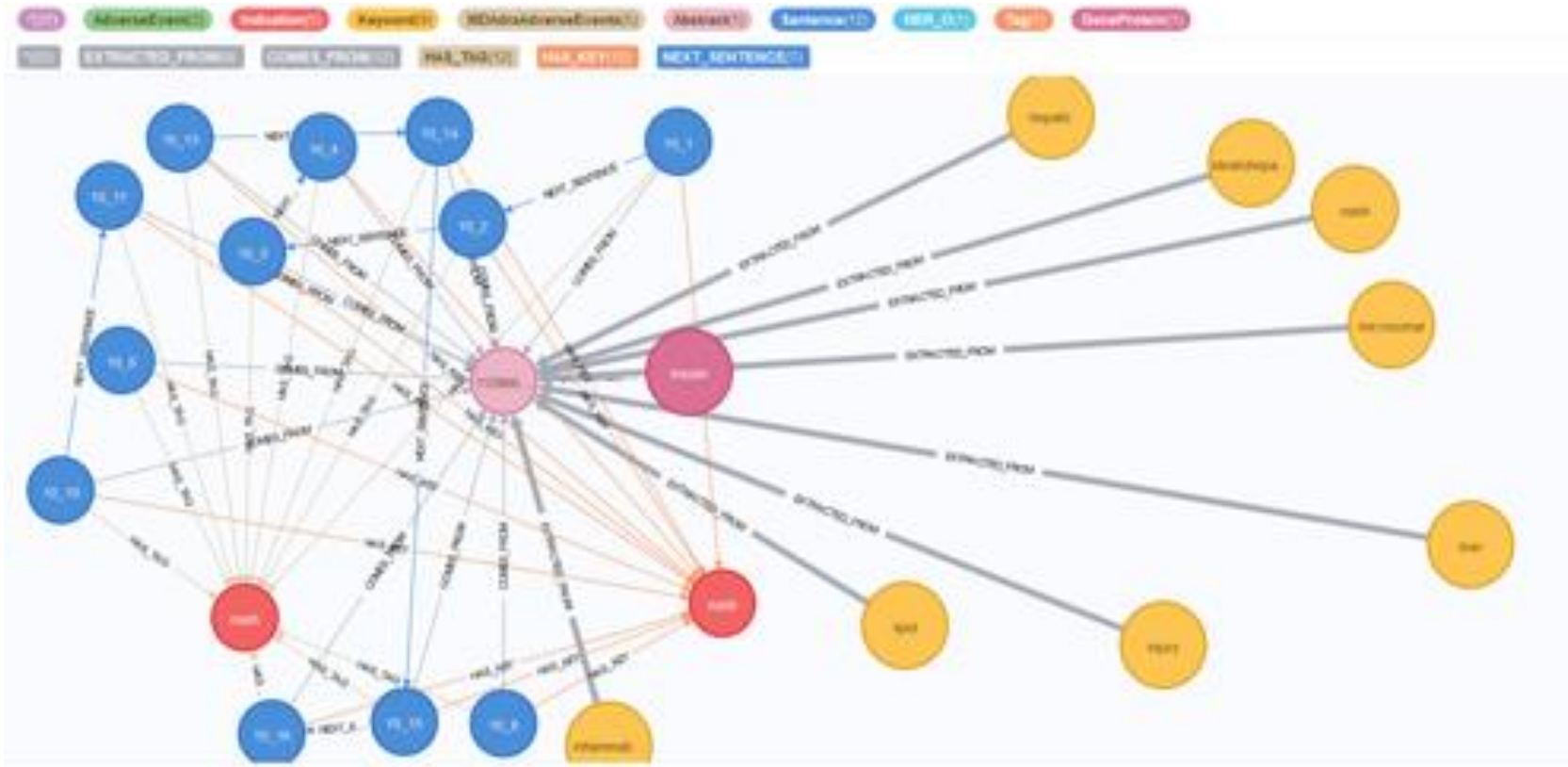
Insights + Results

- Queries focused on NASH, NAFLD, and discovery of related entities
- Looking for communities, paths, and subgraphs
- Weight of relationships

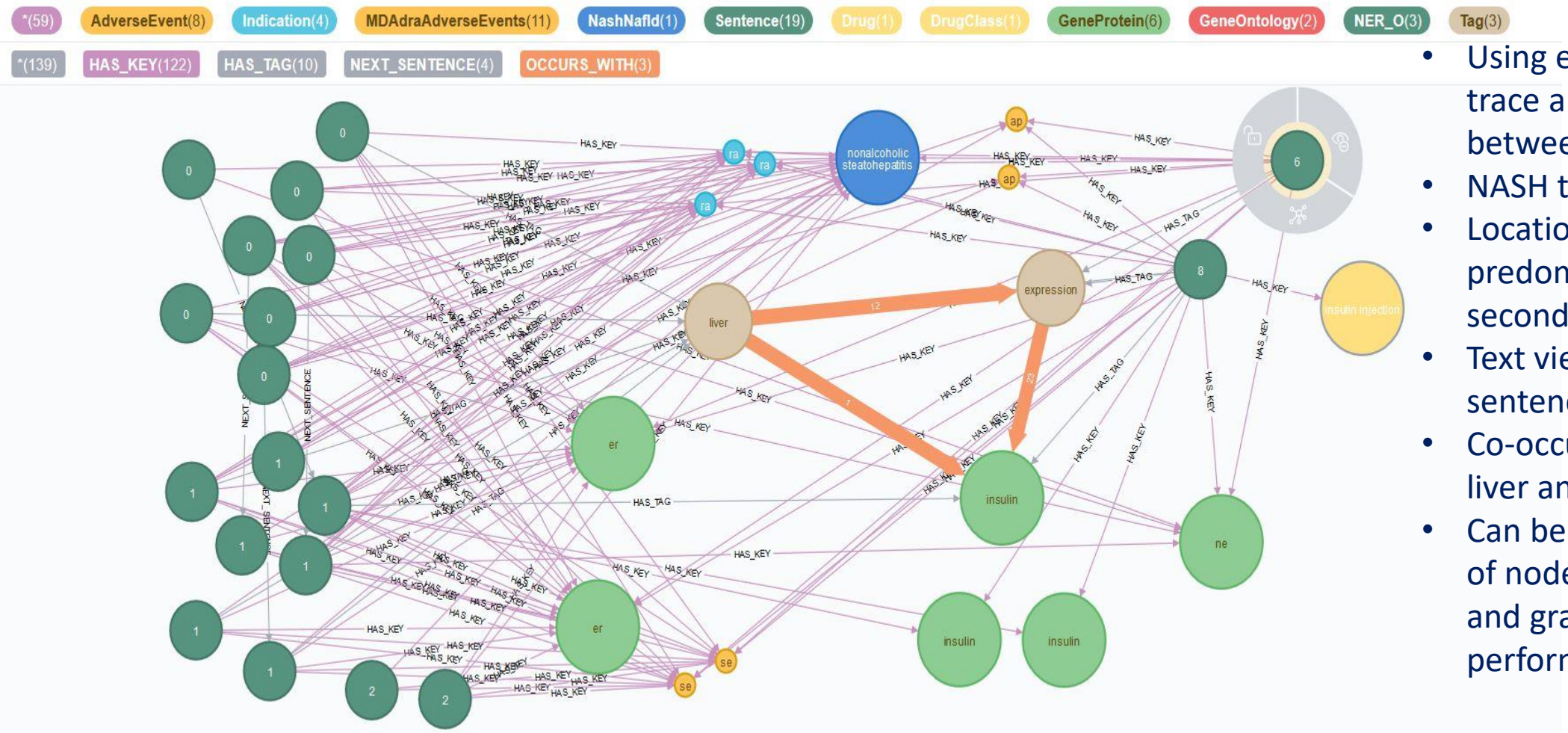


Keyword Extraction

- Keywords extracted from abstracts, along with the sentences.
- In the abstract AND the relationships of those sentences to ‘nash'.



All Shortest Paths-NASH to Insulin Injection

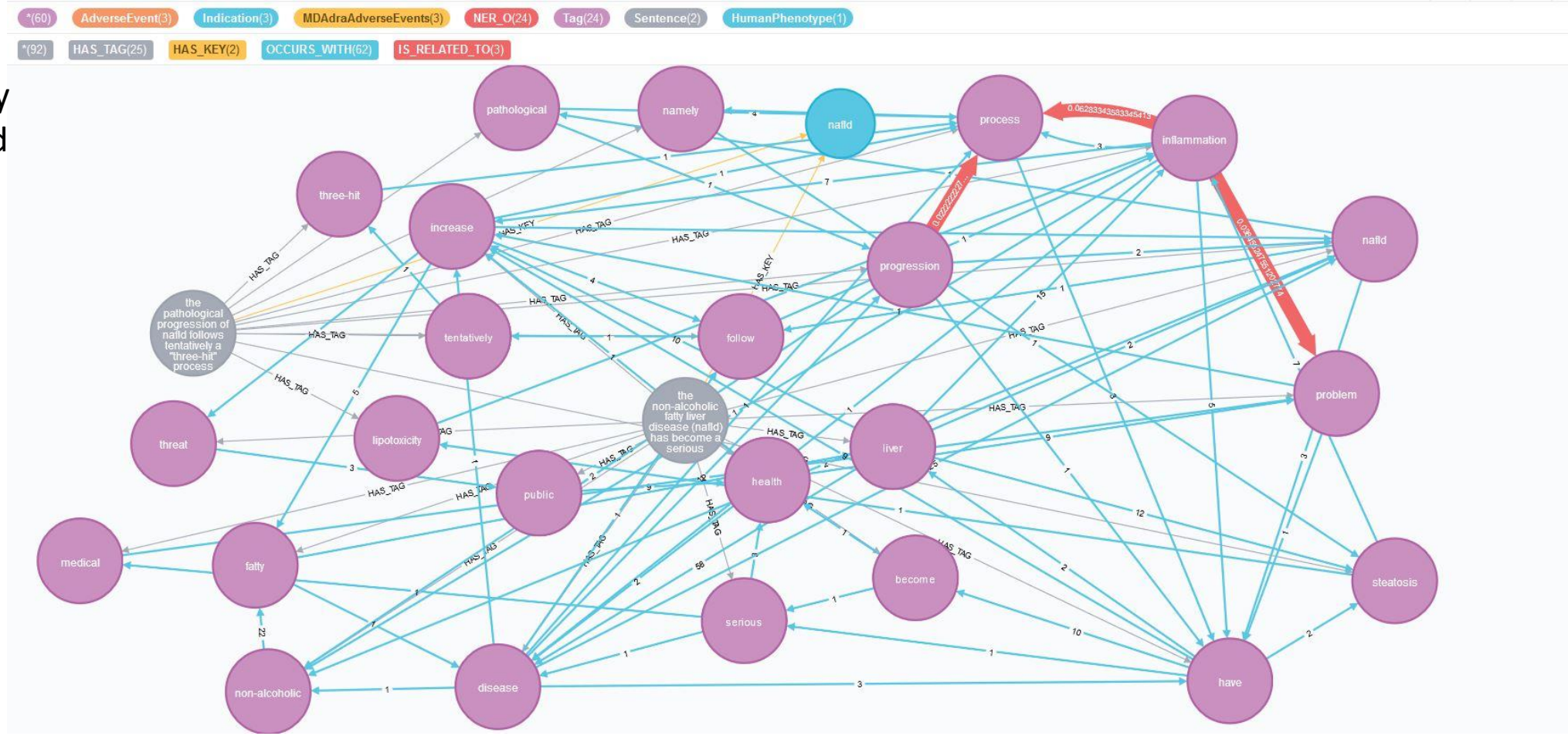


- Using exact node ID's we can trace all of the shortest paths between two nodes
- NASH to Insulin Injection
- Location of NASH key predominately in first and second sentences in paths
- Text viewable from the 7th sentence of an abstract
- Co-occurrences between liver and insulin
- Can be repeated for any pair of nodes-computation power and graph structure impact performance

conversely, nonalcoholic steatohepatitis-associated degs coded for epidermal growth factor receptor and tlr4 signaling with decreased expression of the gtpase rab5 and the lipid phosphohydrolase ppap2b thus highlighting adaptive responses to inflammation, ldl-mediated endocytosis and lipogenesis, respectively.

NAFLD Community Members

- Community may be too large and need to reset



Community Detection-Label Propagation Algorithm

- LPA is used to detect labels becoming too dominant in a densely connected group of nodes
- Appropriate for Drugs-has been used to estimate potentially dangerous combinations of drugs
- Drugs(lp) converged under “label” belong to same community under LPA
- Label 28467 populated by phyto’s and flavonoids
- Label 23646 comprised of Type 2 Diabetes drugs/compounds and insulin desensitizers
- Ciglitazone

label	lp
28467	["morin", "naringenin", "kaempferol", "resveratrol", "isoflavone", "phytoestrogen", "daidzein"]
23646	["thiazolidinedione", "troglitazone", "rosiglitazone", "ciglitazone", "pioglitazone"]
120268	["phenothiazine", "chlorpromazine", "trifluoperazine", "fluphenazine"]
234294	["parthenolide", "nerolidol", "sesquiterpene"]
288948	["epicatechin", "catechin"]
214327	["xanthine", "isobutylmethylxanthine"]
215495	["nobiletin", "flavone"]
109504	["rad001"]
109505	["mk-2206"]
109637	["mk2206"]
110658	["3-isobutyl-1-methylxanthine"]
110661	["dexamethasone"]
111220	["salinomycin"]
111363	["melatonin"]
111577	["mitotane"]

Moving Forward

- We believe Neo4j's architecture, with plugins, has potential as a database to build a Knowledge Graph on top of
- Cypher's (NoSQL language used in) offers incredible flexibility and enables an iterative construction process, but comes with limitations as a standalone NLP and text mining tool
- The Python driver seems like a possible solution to rectify, and we advise attempted implementation for similar projects
- Neo4j offers a cloud product, Aura, and has major provider cloud computing integration-these would significantly speed up build, refactoring, and processing power
- Proactively labeling, organizing, and shuffling the representation of your data is a way to become familiar both with it and Neo4j
- Initial steep learning curve with platform and Cypher that flattens at a point
- Sufficient resources online, emerging technology, and growing community

Appendix

```
//      Create Schema
CALL ga.nlp.createSchema();

//      Set Language to English
CALL ga.nlp.config.setDefaultLanguage('en');

//      Set Pipeline
CALL ga.nlp.processor.addPipeline({textProcessor: 'com.graphaware.nlp.processor.stanford.StanfordTextProcessor', name: 'Processor', processingSteps: {tokenize: true, ner: true, dependency: false}, stopWords: '+,result, all, during',
threadNumber: 20});

//      Call Pipeline
CALL ga.nlp.processor.pipeline.default('Processor');

////////////---------------------------------
//Load data
//      Load JSON Abstract Data:
CALL apoc.load.json("file:///finalabstracts.jsonl") YIELD value AS row
CREATE (n:Abstract {id: row.id, text:row.text});

//      Load Keys and Ontology
LOAD CSV WITH HEADERS FROM "file:///ont4.csv" AS row
CREATE (n:Keys{class: row.class, value:row.value});

//      Indexes Abstracts
CREATE INDEX ON :Abstract(id);

//      Indexes Keys
CREATE INDEX ON :Keys(value);


//      Creates /annotates
MATCH (a:Abstract)
CALL ga.nlp.annotate({text: a.text, id: id(a)})
YIELD result
MERGE (a)-[:CONTAINS_TEXT]->(result)
RETURN count(result);

//      Connects Abstracts and Sentences
MATCH(a:Abstract),(s:Sentence)
WHERE a.text CONTAINS s.text
MERGE (s)-[r:COMES_FROM]->(a) RETURN count(a);

//  Connects Key Occurance in Sentences
MATCH(k:Keys),(s:Sentence)
WHERE s.text CONTAINS k.value
MERGE(s)-[:HAS_KEY]->(k);

```

Appendix.2

```
//------------------------------------------------------------------------
//      Creates first sentence property in abstract
MATCH (s:Sentence)-[r:COMES_FROM]->(a:Abstract)
WHERE s.sentenceNumber=0
SET r.starts='True';

MATCH (s:Sentence)-[r:COMES_FROM]->(a:Abstract)
WHERE NOT s.sentenceNumber=0
SET r.starts='False';

//      Clean up sentence "3)" and add to begining of next sentence
MATCH (s1:Sentence)-[:NEXT_SENTENCE]->(s2:Sentence)-[:NEXT_SENTENCE]->(s3:Sentence)
WHERE s1.id='340_5' AND s2.id='340_6' AND s3.id='340_7'
MERGE (s1)-[:NEXT_SENTENCE]->(s3)

MATCH (s1:Sentence { id: '340_6' })
DETACH DELETE s1;

MATCH(s:Sentence{id:'340_7'})
SET s.text='3) cocs were matured in serum-free medium containing 1 mg/ml polyvinyl alcohol and 0, 10, 100, or 1000 ng/ml leptin (l0, l10, l100, and l1000, respectively), or in medium supplemented with 10% fetal calf serum (fcs) as a positive control.';

MATCH(s:Sentence{id:'340_7'}) SET: s.id:'340_6',s.sentenceNumber:'6';
MATCH(s:Sentence{id:'340_8'}) SET: {s.id:'340_7',s.sentenceNumber:'7'}; MATCH(s:Sentence{id:'340_9'}) SET:{s.id:'340_8',s.sentenceNumber:'8'}; MATCH(s:Sentence{id:'340_10'}) SET:{s.id:'340_9',s.sentenceNumber:'9'}; MATCH(s:Sentence{id:'340_11'}) SET:
{s.id:'340_10',s.sentenceNumber:'10'}; MATCH(s:Sentence{id:'340_12'}) SET: {s.id:'340_11',s.sentenceNumber:'11'}; MATCH(s:Sentence{id:'340_13'}) SET: {s.id:'340_12',s.sentenceNumber:'12'}; MATCH(s:Sentence{id:'340_14'}) SET: {s.id:'340_13',s.sentenceNumber:'13'};

//-------------------------------------------------------------------------------------

//          Creates NE hierarchy
MATCH (a:AnnotatedText)-[:CONTAINS_SENTENCE]->(s:Sentence)-[:SENTENCE_TAG_OCCURRENCE]-
>(to:TagOccurrence)-[:TAG_OCCURRENCE_TAG]->(tag)
WHERE tag:NER_O
WITH a, to, tag
ORDER BY s.id, to.startPosition
WITH a, collect(tag) as tags
UNWIND range(0, size(tags) - 2) as i
WITH a, tags[i] as tag1, tags[i+1] as tag2 WHERE tag1 <> tag2
MERGE (tag1)-[r:OCCURS_WITH]-(tag2)
ON CREATE SET r.freq = 1
ON MATCH SET r.freq = r.freq + 1;


//      Enrichment

MATCH (n:Tag)
CALL ga.nlp.enrich.concept({enricher: 'conceptnet5', tag: n, depth:1, admittedRelationships:['IsA','PartOf','HasA','HasSubevent', 'ReceivesAction', 'DistinctFrom','DerivedFrom','DefinedAs','UsedFor', 'Synonym', 'CapableOf', 'Causes', 'CreatedBy', 'ObstructedBy', 'HasContext',
'MadeOf'], relationshipType:['IsA', 'PartOf','HasA','HasSubevent','ReceivesAction','DistinctFrom','DerivedFrom','DefinedAs', 'UsedFor', 'Synonym', 'CapableOf', 'Causes', 'CreatedBy', 'ObstructedBy', 'HasContext', 'MadeOf']})
YIELD result
SET n:ConceptProcessed
```

Appendix.3

```
MATCH (n:Tag)
CALL ga.nlp.enrich.concept({enricher: 'microsoft', tag: n, depth:1, checkLanguage:false})
YIELD result
SET n:ConceptProcessed
RETURN count(*);

//---------------------------------------------------------------


//Keyword Extraction

MATCH (a:AnnotatedText)
CALL ga.nlp.ml.textRank({annotatedText: a, useDependencies: true})
YIELD result RETURN result;

CREATE INDEX ON :Keyword(value);

CALL ga.nlp.ml.textRank.postprocess({keywordLabel: "Keyword", method: "subgroups"})
YIELD result
RETURN result

CALL apoc.periodic.iterate(
'MATCH (n:AnnotatedText) RETURN n',
'CALL ga.nlp.ml.textRank.postprocess({annotatedText: n, method:"subgroups"}) YIELD result RETURN count(n)',
{batchSize: 100, iterateList:false}
);
//---------------------------------------------------------------
//      Connects Abstracts and Sentences
MATCH(a:Abstract),(s:Sentence)
WHERE a.text CONTAINS s.text
MERGE (s)-[r:COMES_FROM]->(a) RETURN count(a);

//  Connects Key Occurance in Sentences
MATCH(k:Keys),(s:Sentence)
WHERE s.text CONTAINS k.value
MERGE(s)-[:HAS_KEY]->(k);

//---------------------------------------------------------------------------
//      Creates first sentence property in abstract
MATCH (s:Sentence)-[r:COMES_FROM]->(a:Abstract)
WHERE s.sentenceNumber=0
SET r.starts='True';

MATCH (s:Sentence)-[r:COMES_FROM]->(a:Abstract)
WHERE NOT s.sentenceNumber=0
SET r.starts='False';
```

Appendix.4

```
//      Clean up sentence "3)" and add to begining of next sentence
MATCH (s1:Sentence)-[:NEXT_SENTENCE]->(s2:Sentence)-[:NEXT_SENTENCE]->(s3:Sentence)
WHERE s1.id='340_5' AND s2.id='340_6' AND s3.id='340_7'
MERGE(s1)-[:NEXT_SENTENCE]->(s3)

MATCH (s1:Sentence { id: '340_6' })
DETACH DELETE s1;

MATCH(s:Sentence{id:'340_7'})
SET s.text='3) cocs were matured in serum-free medium containing 1 mg/ml polyvinyl alcohol and 0, 10, 100, or 1000 ng/ml leptin (l0, l10, l100, and l1000, respectively), or in medium supplemented with 10% fetal calf serum (fcs) as a positive control.';

MATCH(s:Sentence{id:'340_7'}) SET: s.id:'340_6',s.sentenceNumber:'6';
MATCH(s:Sentence{id:'340_8'}) SET: {s.id:'340_7',s.sentenceNumber:'7'}; MATCH(s:Sentence{id:'340_9'}) SET:{s.id:'340_8',s.sentenceNumber:'8'}; MATCH(s:Sentence{id:'340_10'}) SET:{s.id:'340_9',s.sentenceNumber:'9'}; MATCH(s:Sentence{id:'340_11'}) SET:
{s.id:'340_10',s.sentenceNumber:'10'}; MATCH(s:Sentence{id:'340_12'}) SET: {s.id:'340_11',s.sentenceNumber:'11'}; MATCH(s:Sentence{id:'340_13'}) SET: {s.id:'340_12',s.sentenceNumber:'12'}; MATCH(s:Sentence{id:'340_14'}) SET: {s.id:'340_13',s.sentenceNumber:'13'};



//-----------------------------------------------------------------
CREATE CONSTRAINT ON (m:TagOccurence) ASSERT m.value IS UNIQUE;

//      Delete Tag Occurrences
MATCH (m:TagOccurrence)
DETACH DELETE m;

//  cleans up earlier
MATCH (n)-[r:CONTAINS_TEXT]->()
DETACH DELETE n

//      Refactor Labels to Ontology Classes

MATCH(n),(k:Keys{class:'mdaDraAdverseEvents'})
WITH n, k.value AS ont
WHERE ont=n.value
SET n:MDAdraAdverseEvents;

MATCH(n),(k:Keys{class:'adverseEvent'})
WITH n, k.value AS ont
WHERE ont=n.value
SET n:AdverseEvent;

MATCH(n),(k:Keys{class:'drug'})
WITH n, k.value AS ont
WHERE ont=n.value
SET n:Drug;
```

Appendix.5

```
MATCH(n),(k:Keys{class:'drugClass'})
WITH n, k.value AS ont
WHERE ont=n.value
SET n:DrugClass;

MATCH(n),(k:Keys{class:'geneOntology'})
WITH  n, k.value AS ont
WHERE ont=n.value
SET n:GeneOntology;

MATCH(n),(k:Keys{class:'geneProtein'})
WITH  n, k.value AS ont
WHERE n.value=ont
SET n:GeneProtein;

MATCH(n),(k:Keys{class:'humanPhenotype'})
WITH  n, k.value AS ont
WHERE n.value=ont
SET n:HumanPhenotype;

MATCH(n),(k:Keys{class:'indication'})
WITH  n, k.value AS ont
WHERE n.value=ont
SET n:Indication;
--------------------------------------------------------------------------------
//      Create relationships from Conceptnet 5 relations

MATCH(n)-[r:IS_RELATED_TO{type:'Causes'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:CAUSES{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'PartOf'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:PART_OF{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'HasA'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:HAS_A{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);
```

Appendix.6

```
MATCH(n)-[r:IS_RELATED_TO{type:'HasSubevent'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:HAS_SUBEVENT{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'DistinctFrom'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:DISTINCT_FROM{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'DerivedFrom'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:DERIVED_FROM{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'DefinedAs'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:DEFINED_AS{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'Synonym'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:SYNONYM{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'IsA'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:IS_A{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'CapableOf'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:CAPABLE_OF{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'CreatedBy'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:CREATED_BY{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);
```

Appendix.7

```
MATCH(n)-[r:IS_RELATED_TO{type:'UsedFor'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:USED_FOR{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'ReceivesAction'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:RECEIVES_ACTION{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'HasContext'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:HAS_CONTEXT{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);

MATCH(n)-[r:IS_RELATED_TO{type:'ObstructedBy'}]->(m)
WITH n,m,r.source AS sour, r.weight AS wei
WHERE sour='CONCEPT_NET_5'
MERGE (n)-[r2:OBSTRUCTED_BY{source:'CONCEPT_NET_5', weight:wei}]-(m)
RETURN count(r2);


//  Delete the CONCEPTNET5 is related to
MATCH (a)-[rel:IS_RELATED_TO{source:'CONCEPT_NET_5'}]->(m)
WHERE NOT rel.source='MICROSOFT_CONCEPT'
DELETE rel;


//      See where we are after all of this
MATCH(n)
RETURN count(n), labels(n);

MATCH()-[r]-()
RETURN count(r), type(r);

//      Begin moving properties from annotated text to abstracts

MATCH (a:Abstract)<-[:COMES_FROM]-(s:Sentence),(w:AnnotatedText)-[:CONTAINS_SENTENCE]->(s)
WHERE s.sentenceNumber=0
SET a.anntextid=w.id;

MATCH (a:Abstract),(n:AnnotatedText)
WHERE a.anntextid=n.id
SET a.numTerms=n.numTerms;
```

Appendix.8

```
//      recreate keyword extraction to abstracts

MATCH (a:Abstract),(k:Keyword)-[r:DESCRIBES]->(an:AnnotatedText)
WHERE a.anntextid=an.id
MERGE(k)-[d:EXTRACTED_FROM{relevance:r.relevance}]-(a)
RETURN count(d);

//      Delete Annotated text

MATCH (n:AnnotatedText)
DETACH DELETE n;

//-------------------------------------
MATCH(n) REMOVE n.startPosition
CREATE(n:TagOccurence {value:'l'})
MATCH(n) WHERE n.value='l' DETACH DELETE n

DROP INDEX ON :Keys(value)

DROP CONSTRAINT ON ( tagoccurence:TagOccurence ) ASSERT tagoccurence.value IS UNIQUE

MATCH(k:Keys:AdverseEvent)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:GeneProtein)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:Drug)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:DrugClass)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:GeneOntology)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:DrugClass)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:HumanPhenotype)
REMOVE k:Keys
RETURN labels(k);
```

Appendix.9

```
MATCH(k:Keys:Indication)
REMOVE k:Keys
RETURN labels(k);

MATCH(k:Keys:MDAdraAdverseEvents)
REMOVE k:Keys
RETURN labels(k);
```

Appendix2.1

We discussed our concerns about specific location/referential information lost using our build process and potential solutions to the problem. Using our pipeline, 198,630 Nodes and 397,260 (double) new relationships are created during abstract annotation. We able to either capture from, or transfer to, existing nodes, all properties/information less two on nodes labeled (n:TagOccurrence). n.startPosition and n.endPosition-which is location information (in abstract/annotated) and is important. Thought was to use NoSQL and turn occurrences in to allowed uneven property count with SET n.position=(n.endPosition-n.startPosition)/2 and write a condition loop for n.position +1 to the k occurrences and concatenated abstract properties to create a unique key. This would save a lot of screen real estate and optimize visual understandability of query results, but is unrealistic given the frequency of many terms. There was an internal discussion about this and we recalled that we had been instructed to delete "start" and "end" as columns from the Keys/entities data we were given. This sheet was imported with all five columns and

(MATCH(a:Keys),(n:TagOccurencces)
WHERE a.start=n.StartPosition AND a.end=a.EndPosition AND a.value=m.value
RETURN a,n

did not provide results. Finally, we realized that we have been querying imported properties with numbers as text/string format from Keys and Integer properties generated for TagOccurrences. After formatting cells in Excel correctly, the same query returned all properties for (a) and (m), values matching. This confirms that yes, if we found necessary, the Keys' positions can be used for starting and ending nodes in paths after minor refactoring. This also confirmed that [redacted] itself had at minimum already annotated text and had given us the exact information which we have spent(well more time than we are willing to admit). [redacted] obviously knows this and I, Mike, believe discussing this discrepancy, result with the client is important and would be embarrassed by overlooking or ignoring this. This changes our understanding of the project because the company has obviously explored Neo4j with this data set through annotation and a procedure to create Keys. If it was a formatting/import overlook in our base code, I completely understand and only figured this out from 20+ builds. If it was intentional , the gap in communication after confirmation is what has had us scrambling the past 18 hours and any reasoning for this would (be appreciated and) help/have helped us understand the problem much better We rebuilt through enrichment and also cleaned up the build code for ease of replicability be client. Below, although we prefer our database and with time, would implement only a few(mostly for code cleanliness) updates including recapturing position information for Keys.

Appendix2.2

Simplified Code for DB with [redacted]'s exact files

```
// DATA FILES:          1) abstracts.jsonl
//                      2) [redacted].xlsx with following change
//                                  -saved as CSV-UT8
//                                  -Column headers cased to lower for all along with following changes:
-//                                  "DocID" to "abstractId", "EntityType" to "code", "Phrase to "value"

//                      3) [redacted]/xlsx with following change
//                                  -saved as CSV-UT8
//                                  -Column headers cased to lower for all along with following changes:
//                                  "Name" to "values";
//                                  -comments from Column C deleted
```

Appendix2.3

```
//                          Create Schema
CALL ga.nlp.createSchema();
//                          Set Language to English
CALL ga.nlp.config.setDefaultLanguage('en');
//                          Set Pipeline-Named Entity Recognition set to false
CALL ga.nlp.processor.addPipeline({textProcessor: 'com.graphaware.nlp.processor.stanford.StanfordTextProcessor', name: 'Processor',
processingSteps: {tokenize: true, ner: false, dependency: false}, stopWords: '+,result, all, during',
threadNumber: 20});
//                          Call Pipeline
CALL ga.nlp.processor.pipeline.default('Processor');
//          Load data-DATA IS EXACTLY AS GIVEN BY [redacted] HERE
//                          Load JSON Abstract Data:
CALL apoc.load.json("file:///abstracts.jsonl") YIELD value AS row
CREATE (n:Abstract {id: toInteger(row.id), text:row.text});

//  Use APOC procedure to makes all property values lower case for all nodes abstract text is including
MATCH (n)
WITH n, [x IN keys(n)
      WHERE n[x] =~ '.*'
] as props
UNWIND props as p
CALL apoc.create.setProperty(n, p, toLower(n[p])) YIELD node
RETURN node
//                          Load Keys and Ontology  'definitions.csv' can be substituted
LOAD CSV WITH HEADERS FROM "file:///fullontdef.csv" AS row
CREATE (n:Definitions{code: row.code, value:row.value});

//                          Load Keys and Ontology -we change from varchar/text to Integers during import with
//                          Cypher here instead of in Excel/Python
LOAD CSV WITH HEADERS FROM "file:///fullkeyont.csv" AS row

CREATE (n:Keys{abstractId: toInteger(row.abstractId), code: row.code, value: row.value, start: toInteger(row.start), end: toInteger(row.end)
});
                                                                    //                  Indexes Abstracts
                                                                    CREATE INDEX ON :Abstract(id);
CREATE INDEX ON :Abstract(id);
//                          Indexes Keys
CREATE INDEX ON :Keys(value);
//                          Creates /annotates Abstracts-we take properties from Created AnnotatedText nodes
//   and set as abstract properties immediately here
MATCH (a:Abstract)
CALL ga.nlp.annotate({text: a.text, id: id(a)})
YIELD result
SET a.atid=result.id, a.numTerms=result.numTerms;
//              Create length of characters property for each abstract to go along wit
MATCH(a:Abstract) SET a.lenChar=length(a.text);
//              This apoc procedure creates label for a node class from properties of node in another label class //in one
procedure, eliminating 8(would be 40 here) previous statements .  Deleting that property could //be set at end like we did in
code for other model but we cannot confirm that no information will be //lost using the apoc procedure.  Simpler and faster, but
we know our way works
MATCH (n:Keys)
CALL apoc.create.addLabels([ id(n) ], [ n.code ])
YIELD node
RETURN node
//                                                  Creates NE hierarchy
MATCH (a:AnnotatedText)-[:CONTAINS_SENTENCE]->(s:Sentence)-[:SENTENCE_TAG_OCCURRENCE]-
>(to:TagOccurrence)-[:TAG_OCCURRENCE_TAG]->(tag)
WHERE tag:NER_O
WITH a, to, tag
ORDER BY s.id, to.startPosition
WITH a, collect(tag) as tags
UNWIND range(0, size(tags) - 2) as i
WITH a, tags[i] as tag1, tags[i+1] as tag2 WHERE tag1 <> tag2
MERGE (tag1)-[r:OCCURS_WITH]-(tag2)
ON CREATE SET r.freq = 1
ON MATCH SET r.freq = r.freq + 1;
//                          Enrichment- ConceptProcessed nodes are not created, only Tags //enriched.
MATCH (n:Tag)
CALL ga.nlp.enrich.concept({enricher: 'conceptnet5', tag: n, depth:1, admittedRelationships:['IsA','PartOf','HasA','HasSubevent',
'ReceivesAction', 'DistinctFrom','DerivedFrom','DefinedAs','UsedFor', 'Synonym', 'CapableOf', 'Causes', 'CreatedBy',
'ObstructedBy', 'HasContext', 'MadeOf'], relationshipType:['IsA',
'PartOf','HasA','HasSubevent','ReceivesAction','DistinctFrom','DerivedFrom','DefinedAs', 'UsedFor', 'Synonym', 'CapableOf',
'Causes', 'CreatedBy', 'ObstructedBy', 'HasContext', 'MadeOf']})
RETURN count(*);
MATCH(n:Tag) CALL ga.nlp.enrich.concept({enricher: 'microsoft', tag: n, depth:1, checkLanguage:false}) YIELD result
RETURN count(*);
```

Appendix2.4

JACCARD SIMILARITY
//see path
MATCH path = (p:Gene_ontology)-[:IS_RELATED_TO]->(label)
RETURN path
LIMIT 25;

//collect similarities and relationship info
MATCH (p:GeneOntology)-[:IS_REALTED_TO]->(label)
WITH {item:id(p), categories: collect(id(label))} as userData

WITH collect(userData) as data

CALL algo.similarity.jaccard(data, {topK: 3, similarityCutoff: 0.9, write: true})

YIELD p25, p50, p90, p99, p999, p100, write
RETURN p25, p50, p90, p99, p999, p100, write

//see patb
MATCH path = (p1:GeneOntology)-[r:SIMILAR]->(p2:GeneOntology)
RETURN path
LIMIT 20

LPA/Similarity Results

//CALL algo.labelPropagation("GeneOntology", "SIMILAR","BOTH")

MATCH (p:GeneOntology)
RETURN p.partition AS partition, count(*) AS count
ORDER BY count DESC
LIMIT 10

MATCH path = (p:Gene_ontology)-[:IS_RELATED_TO]->(label)
RETURN path
LIMIT 25;

// Creates path from -NAFLD nodes to a node (m)...a sentence) to non-acronym NASH, and keys also in that sentence
MATCH p=({value:'nafld'}0-[]-(m)-[]-({value:'nonalcoholic steatohepatitis}),(m)-[:HAS_KEY]-(j)
RETURN p,m, LIMIT 25

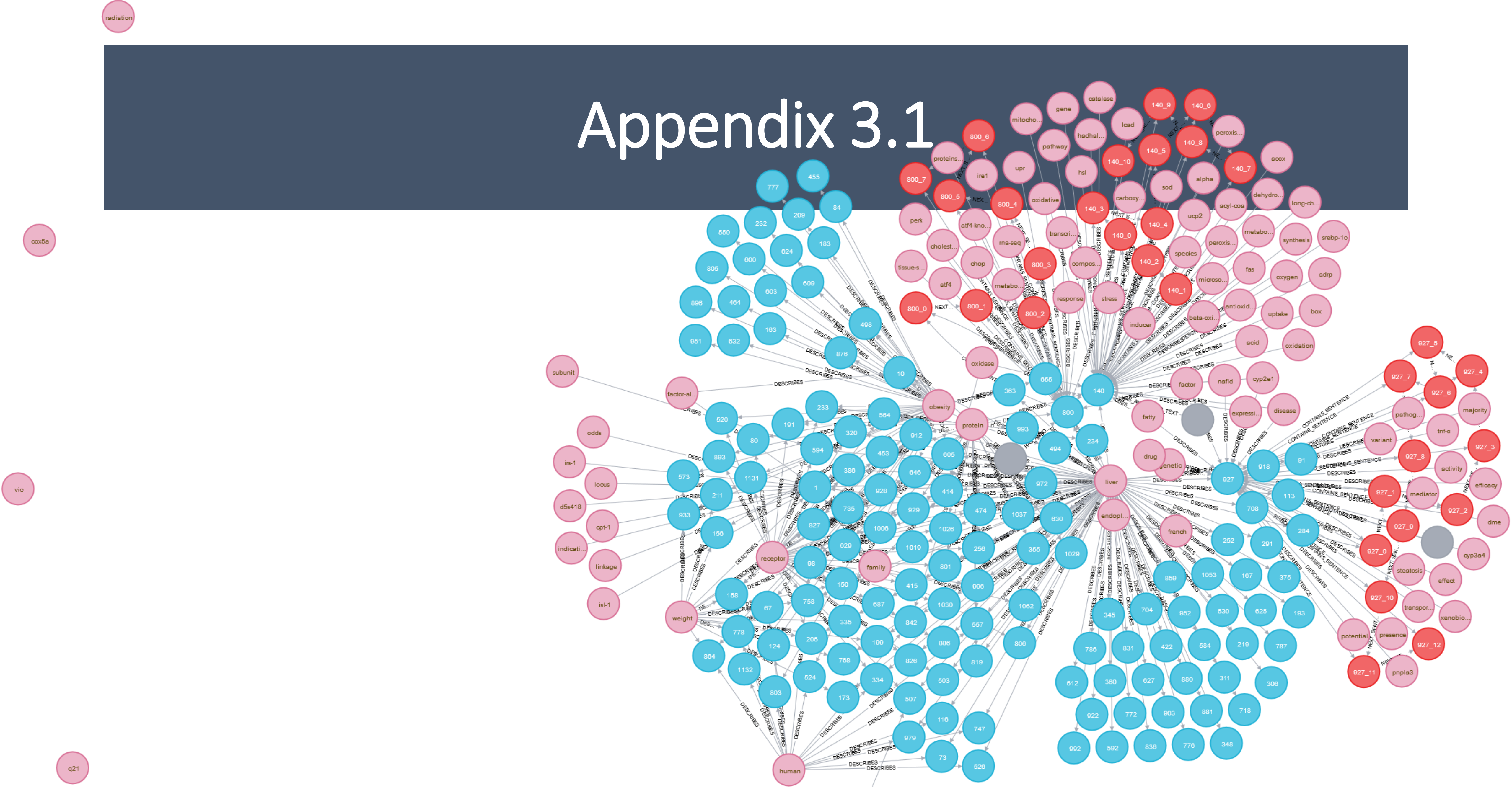
// Observe Community Members of community 12005 created earlier that are two hops(of any relationships) away from NAFLD nodes
MATCH p=(n)-{*2]-(value:'nafld’})
WHERE n.community=12005
RETURN p LIMIT 25

//LPA
CALL algo.labelPropagation.stream("Drug", "IS_RELATED_TO",
{ iterations: 10 })
YIELD nodeId, label
RETURN label,
collect(algo.getNodeById(nodeId).value) AS lp ORDER BY size(lp) DESC

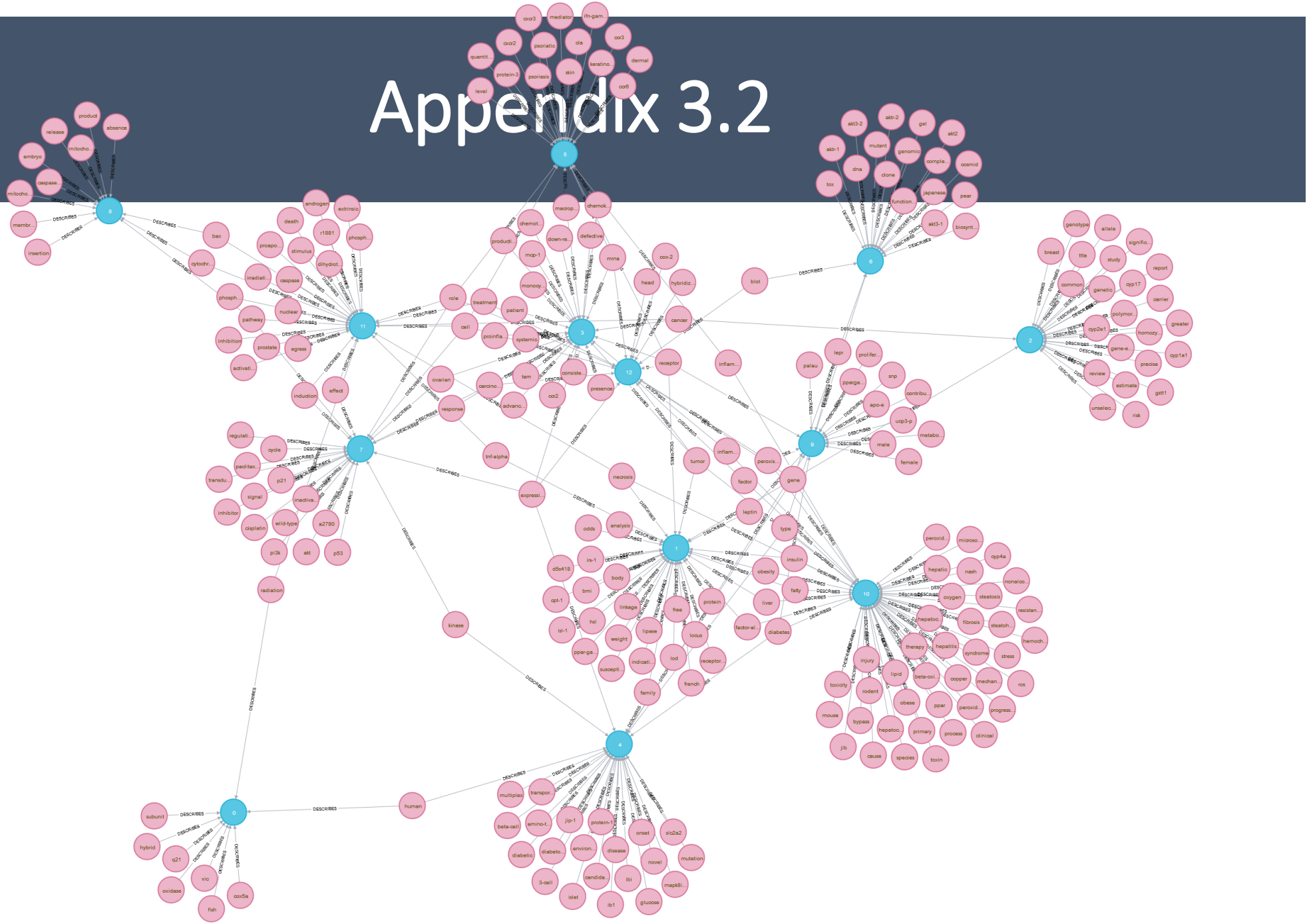
//ALL SHORTEST APTHS
MATCH(n:NashNafld),(m:DrugClass) WHERE ID(n)=1841 AND ID(m)=2543
RETURN allShortestPaths((n)-[*]-(m)) LIMIT 30

//Slide 15
MATCH p=(n:NashNafld)-[*2..3]-()
RETURN p LIMIT 25

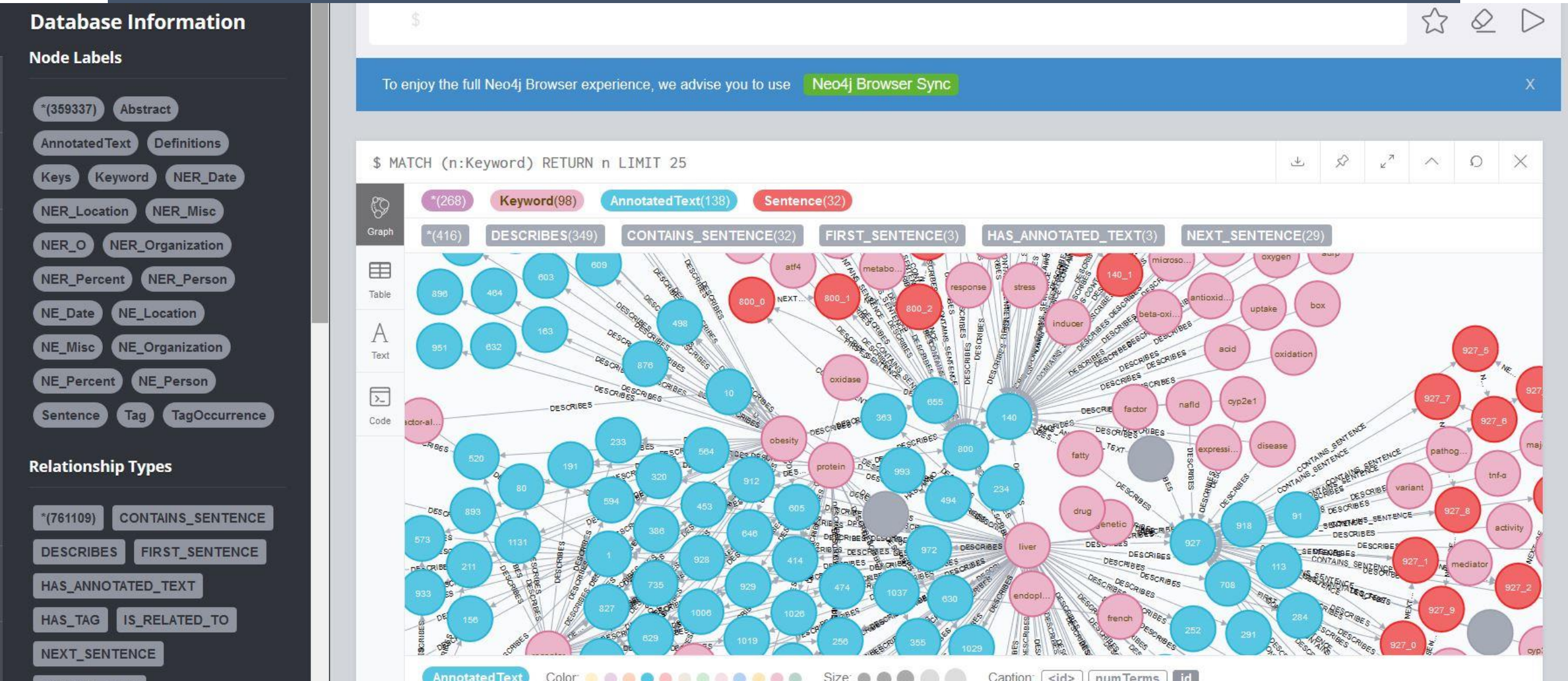
Appendix 3.1


radiation
cox5a
vic
q21
subunit
factor-al...
odds
irs-1
locus
d5s418
cpt-1
indicati...
linkage
isl-1
weight
human
receptor
family
obesity
protein
liver
endopl...
french
drug
genetic
fatty
factor
nafld
cyp2e1
disease
expressi...
oxidase
inducer
stress
response
compos...
transcri...
oxidative
carboxy...
sod
ucp2
species
microso...
antioxid...
beta-oxi...
acid
oxidation
uptake
box
oxygen
adrp
fas
synthesis
srebp-1c
metabo...
peroxis...
dehydro...
long-ch...
acyl-coa
alpha
acox
catalase
lcad
hadhal...
gene
mitocho...
pathway
hsl
upr
ire1
proteins...
perk
atf4-kno...
rna-seq
cholest...
chop
tissue-s...
atf4
pathog...
variant
tnf-α
majority
activity
efficacy
mediator
dme
cyp3a4
steatosis
effect
transpor...
xenobio...
presence
potential
pnpla3
140_0
140_1
140_2
140_3
140_4
140_5
140_6
140_7
140_8
140_9
140_10
800_0
800_1
800_2
800_3
800_4
800_5
800_6
800_7
927_0
927_1
927_2
927_3
927_4
927_5
927_6
927_7
927_8
927_9
927_10
927_11
927_12
DESCRIBES
CONTAINS_SENTENCE
NEXT_SENTENCE

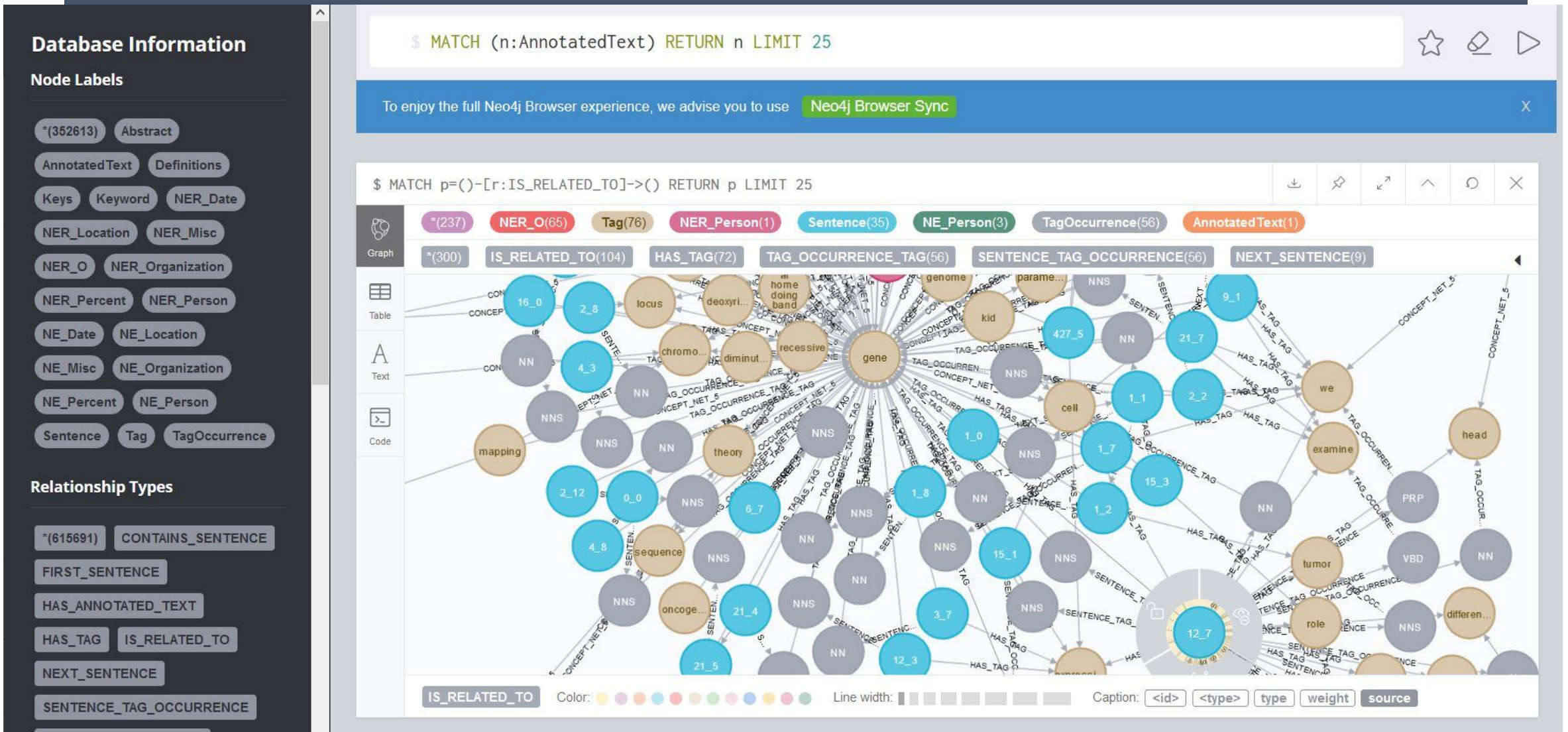
Appendix 3.2



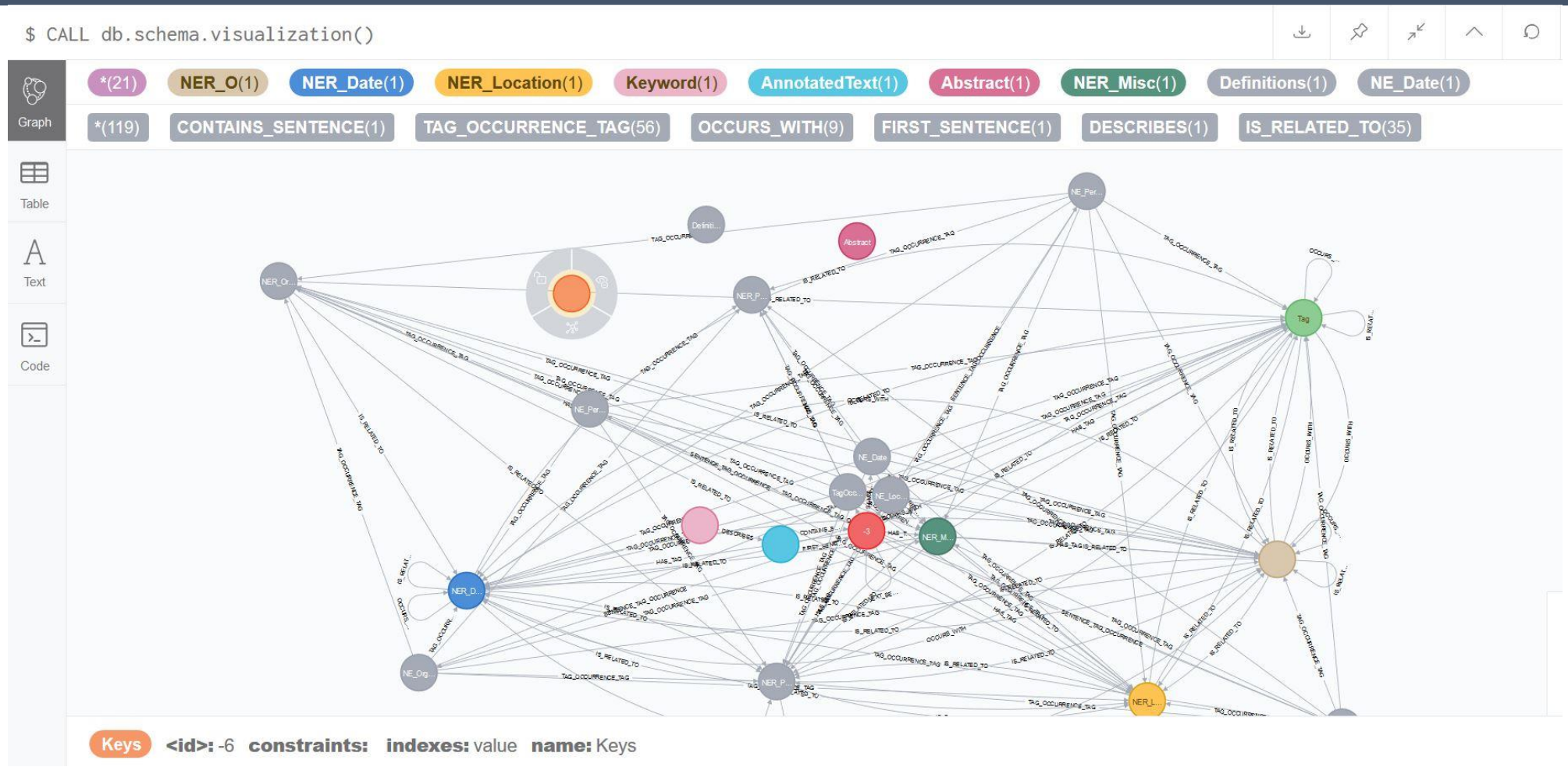
Appendix 3.3


Database Information
Node Labels
*(359337)
Abstract
AnnotatedText
Definitions
Keys
Keyword
NER_Date
NER_Location
NER_Misc
NER_O
NER_Organization
NER_Percent
NER_Person
NE_Date
NE_Location
NE_Misc
NE_Organization
NE_Percent
NE_Person
Sentence
Tag
TagOccurrence
Relationship Types
*(761109)
CONTAINS_SENTENCE
DESCRIBES
FIRST_SENTENCE
HAS_ANNOTATED_TEXT
HAS_TAG
IS_RELATED_TO
NEXT_SENTENCE
To enjoy the full Neo4j Browser experience, we advise you to use
Neo4j Browser Sync
$ MATCH (n:Keyword) RETURN n LIMIT 25
Graph
Table
Text
Code
*(268)
Keyword(98)
AnnotatedText(138)
Sentence(32)
*(416)
DESCRIBES(349)
CONTAINS_SENTENCE(32)
FIRST_SENTENCE(3)
HAS_ANNOTATED_TEXT(3)
NEXT_SENTENCE(29)
AnnotatedText
Color:
Size:
Caption:
<id>
numTerms
id

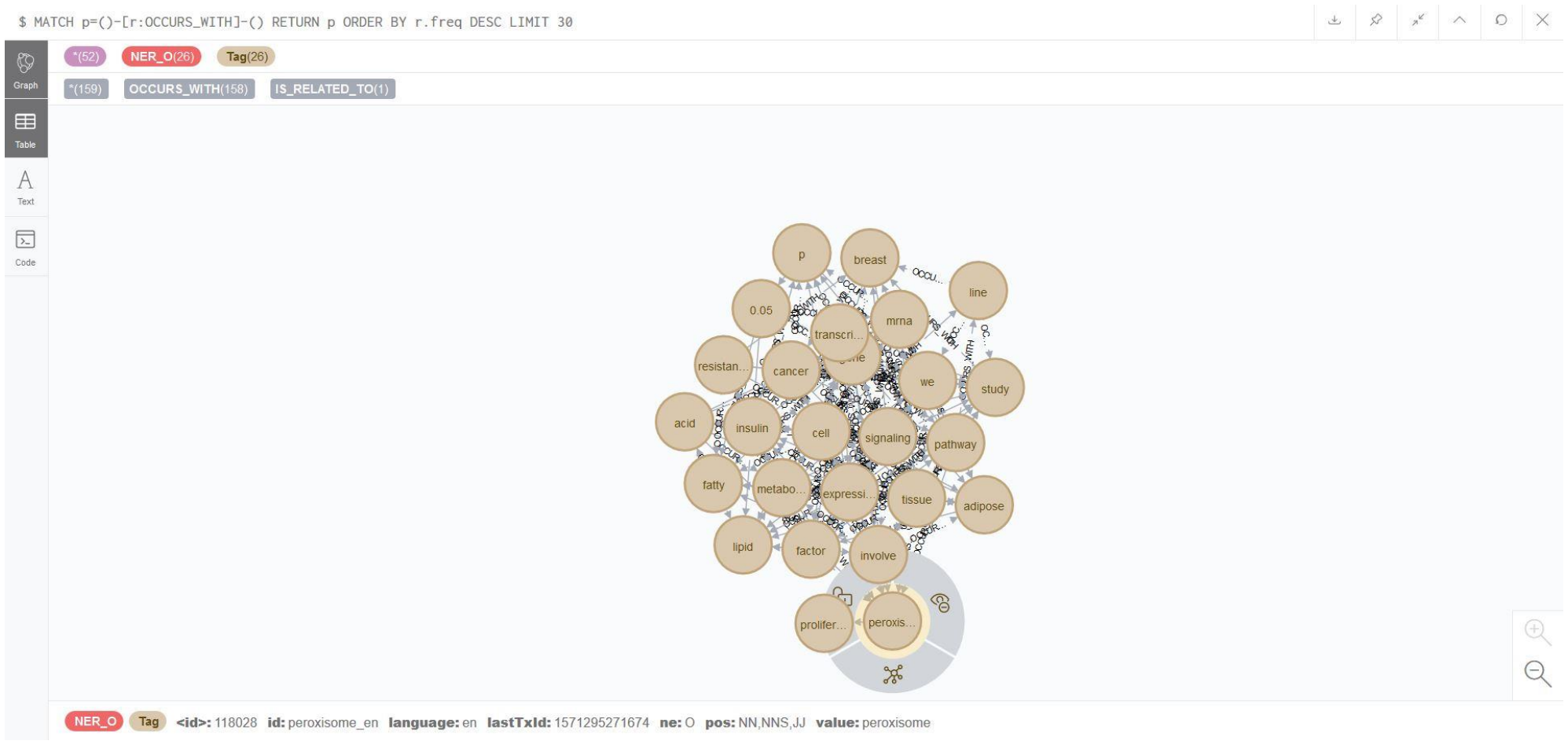
Appendix 3.4


Database Information
Node Labels
*(352613)
Abstract
AnnotatedText
Definitions
Keys
Keyword
NER_Date
NER_Location
NER_Misc
NER_O
NER_Organization
NER_Percent
NER_Person
NE_Date
NE_Location
NE_Misc
NE_Organization
NE_Percent
NE_Person
Sentence
Tag
TagOccurrence
Relationship Types
*(615691)
CONTAINS_SENTENCE
FIRST_SENTENCE
HAS_ANNOTATED_TEXT
HAS_TAG
IS_RELATED_TO
NEXT_SENTENCE
SENTENCE_TAG_OCCURRENCE
$ MATCH (n:AnnotatedText) RETURN n LIMIT 25
To enjoy the full Neo4j Browser experience, we advise you to use
Neo4j Browser Sync
$ MATCH p=()-[r:IS_RELATED_TO]->() RETURN p LIMIT 25
Graph
Table
Text
Code
*(237)
NER_O(65)
Tag(76)
NER_Person(1)
Sentence(35)
NE_Person(3)
TagOccurrence(56)
AnnotatedText(1)
*(300)
IS_RELATED_TO(104)
HAS_TAG(72)
TAG_OCCURRENCE_TAG(56)
SENTENCE_TAG_OCCURRENCE(56)
NEXT_SENTENCE(9)
gene
mapping
theory
sequence
oncoge...
locus
chromo...
diminut...
recessive
kid
cell
we
examine
head
PRP
VBD
NN
tumor
role
differen...
IS_RELATED_TO
Color:
Line width:
Caption:
<id>
<type>
type
weight
source

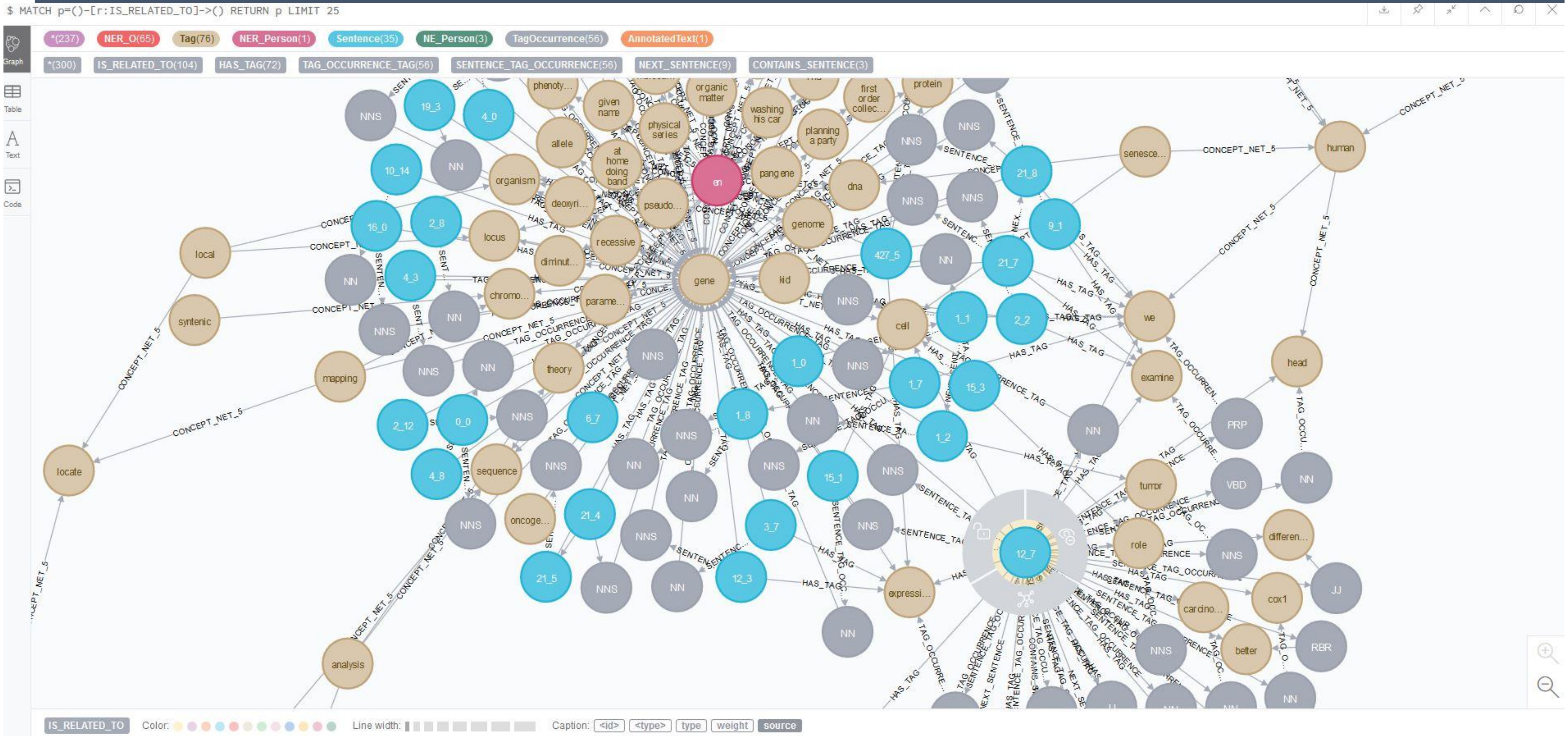
Appendix 3.5


$ CALL db.schema.visualization()
Graph
Table
Text
Code
*(21)
NER_O(1)
NER_Date(1)
NER_Location(1)
Keyword(1)
AnnotatedText(1)
Abstract(1)
NER_Misc(1)
Definitions(1)
NE_Date(1)
*(119)
CONTAINS_SENTENCE(1)
TAG_OCCURRENCE_TAG(56)
OCCURS_WITH(9)
FIRST_SENTENCE(1)
DESCRIBES(1)
IS_RELATED_TO(35)
Keys
<id>: -6
constraints:
indexes: value
name: Keys

Appendix 3.6


$ MATCH p=()-[r:OCCURS_WITH]-() RETURN p ORDER BY r.freq DESC LIMIT 30
Graph
Table
Text
Code
*(52)
NER_O(26)
Tag(26)
*(159)
OCCURS_WITH(158)
IS_RELATED_TO(1)
p
breast
line
0.05
mrna
transcri...
resistan...
cancer
we
study
acid
insulin
cell
signaling
pathway
fatty
metabo...
expressi...
tissue
adipose
lipid
factor
involve
prolifer...
peroxis...
NER_O
Tag
<id>: 118028
id: peroxisome_en
language: en
lastTxId: 1571295271674
ne: O
pos: NN,NNS,JJ
value: peroxisome

Appendix 3.7


$ MATCH p=()-[r:IS_RELATED_TO]->() RETURN p LIMIT 25
*(237)
NER_O(65)
Tag(76)
NER_Person(1)
Sentence(35)
NE_Person(3)
TagOccurrence(56)
AnnotatedText(1)
*(300)
IS_RELATED_TO(104)
HAS_TAG(72)
TAG_OCCURRENCE_TAG(56)
SENTENCE_TAG_OCCURRENCE(56)
NEXT_SENTENCE(9)
CONTAINS_SENTENCE(3)
Graph
Table
Text
Code
gene
en
human
head
local
syntenic
mapping
locate
analysis
organism
allele
locus
recessive
theory
sequence
genome
dna
cell
we
examine
tumor
role
better
cox1
protein
IS_RELATED_TO
Color:
Line width:
Caption:
<id>
<type>
type
weight
source

Thank You

&

Questions