Big data (and metadata)

In big data waves, it is difficult to see a particular section or find a particular drop.
An issue for knowledge organization

Structured data

Gaps between large data sets and knowledge that can be viewed, interacted, and acted upon

Structured knowledge

Data to knowledge

NKOS workshop at ICADL 2015
Graph database as a solution

• Relational DBs
  – Not designed to cope with the scale and agility challenges
  – Not built to take advantage of cheap storage and processing power available today
  – Require predefined data schemas
  – Usually scale vertically

• Graph database
  – One of the NoSQL DB types
  – Similar to the abstract model of subject-predicate-object
  – Built to allow the insertion of data without a predefined schema
Graph structures

• Contain:
  – **Nodes** represent entities such as people, organizations, or things
  – **Properties** are pertinent information that relate to nodes.
    • i.e. If “iSchool” was a node, it might have properties such as “college”, “Syracuse University”, and “degree program”
  – **Edges** are the lines that connect nodes to nodes or nodes to properties. They represent the relationship between the two. Most of the important information is stored in the edges.
Graph Stores

• Provide index-free adjacency, meaning that every element contains a direct pointer to its adjacent elements and no index lookups are necessary
  – Graph queries largely involve using this locality to traverse through the graph, literally chasing pointers
  – Operations can be carried out with extreme efficiency, traversing millions of nodes per second
Graph Stores

• Place a heavy emphasis on the *relationships* between data objects and are designed to store *interconnected data*
An Example: Converting MySQL To Neo4j

Why do we do it?
How does it work?
How is it related to knowledge organization?
# The MySQL database

Size of the data:

<table>
<thead>
<tr>
<th>Property</th>
<th>Overall network</th>
<th>Submission network</th>
<th>Publication network</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total references</td>
<td>1,360,938</td>
<td>1,015,697</td>
<td>345,241</td>
</tr>
<tr>
<td>Total vertices</td>
<td>531,019</td>
<td>386,133</td>
<td>404,466</td>
</tr>
<tr>
<td>Total edges</td>
<td>121,471,078</td>
<td>101,305,810</td>
<td>9,909,522</td>
</tr>
<tr>
<td>Clusters</td>
<td>2,699</td>
<td>4860</td>
<td>1487</td>
</tr>
</tbody>
</table>

Number of annotations by organism

<table>
<thead>
<tr>
<th>ID</th>
<th>Class name</th>
<th>Parent name</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>9606</td>
<td>Homo sapiens</td>
<td>Homo</td>
<td>2,039,8,647</td>
</tr>
<tr>
<td>10090</td>
<td>Mus musculus</td>
<td>Mus</td>
<td>9,789,456</td>
</tr>
<tr>
<td>408172</td>
<td>Marine metagenome</td>
<td>Ecological metagenomes</td>
<td>6,261,089</td>
</tr>
<tr>
<td>32630</td>
<td>Synthetic construct</td>
<td>Artificial sequences</td>
<td>4,475,898</td>
</tr>
<tr>
<td>4577</td>
<td>Zea mays</td>
<td>Zea</td>
<td>3,953,008</td>
</tr>
<tr>
<td>9823</td>
<td>Sus scrofa</td>
<td>Sus</td>
<td>3,304,324</td>
</tr>
<tr>
<td>77133</td>
<td>Uncultured bacterium</td>
<td>Environmental samples</td>
<td>3,080,129</td>
</tr>
<tr>
<td>3702</td>
<td>Arabidopsis thaliana</td>
<td>Arabidopsis</td>
<td>2,337,308</td>
</tr>
<tr>
<td>9913</td>
<td>Bos taurus</td>
<td>Bos</td>
<td>2,209,082</td>
</tr>
</tbody>
</table>
It is computationally expensive and slow in response time in querying the data due to the very large size of the tables.
Purpose for migrating from a relational DB to a graph DB

• Avoid too many inner joins in relational database to improve query performance
• Develop data products by embedding predefined queries
• Visualize query results in real time
• Experiment with converting relational data model into a model suitable for Linked Data applications
Strategies

• Popular method: export data as a CSV and import into Neo4j using independent Batch Importer

• To change from relational to graph, data structure may need to be slightly altered
  – Need to create a model to clarify what are entities, relationships and properties
  – Focused on the specific use case
  – Get data from rows and columns to nodes and edges
Developing models

Scientist → coauthored → Scientist
Scientist → authored → Reference
Reference → associated_with → Annotation
Reference → is_about → Taxonomy
Taxonomy → parent_of → Division
Annotation → is_about → Taxonomy
Division → parent_of → Taxonomy
Steps in converting data

• Converting data in MySQL tables into CSV files
• Write Cypher queries to import data into Neo4j

Sample code for setting up the Neo4j server

```sql
#Annotation
using periodic commit 1000
load csv from
'file:/home/neo4j/Annotation_1.csv' as row
fieldterminator ';
create (:Annotation {gi: toInt(row[0]),
version: toInt(row[1]),
accession: row[2],
definition: row[3],
segment: row[4],
locus_name: row[5],
locus_sequence_length: row[6],
locus_sequence_strands: row[7],
locus_nucleic_acid_type: row[8],
locus_linear_circular: row[9],
locus_date: row[10]});
```
Neo4j 2.2.2

Node labels
- Annotation
- Consortium
- Division
- Reference
- Scientist
- Taxonomy

Relationship types
- associated_with
- authored
- coauthor
- is_about
- part_of

Property keys
- accession
- authId
- consortium
- count
- definition
- divCode
- divId
- divName
- gt
- journal
- locus_date
- locus_linear_circular
- locus_name
- locus_nucleic_acid_type
- locus_sequence_length
- locus_sequence_strands
- name
- pubmed
- rank
- reference
- refId

MATCH (a:Taxonomy), (b:Annotation) MERGE (a)-[:is_about]->(b) return * LIMIT 100;

Displaying 101 nodes, 102 relationships (completed with 102 additional relationships).
Sample query

MATCH (a:Scientist), (b:Scientist) MERGE a-[r:coauthor]->(b) RETURN * LIMIT 125
Sample query
Sample query

MATCH ()-[r]->() RETURN r LIMIT 125
Summary

• Graph databases
  – Useful for presenting relationships/networks
  – Scalable
  – Allow for use together with R, Python, and other languages

• Data structure resembles RDF triples
Conclusion

• Knowledge organization in data-driven environment
  – Goes beyond controlled vocabularies and classification schemes
  – Fills the gap between data and knowledge
  – Models data into the structures for linkable data sets and real-time interaction between users and data as well as between computers
Sample queries

MATCH (a:Scientist), (b:Scientist) MERGE a-[r:coauthor]->(b) RETURN * LIMIT 25

MATCH ()-[r]->() RETURN r LIMIT 125

MATCH (a:Taxonomy), (b:Annotation) MERGE (a)-[:is_about]->(b) return * LIMIT 50

http://neo4j-genbank.syr.edu:7474/browser/