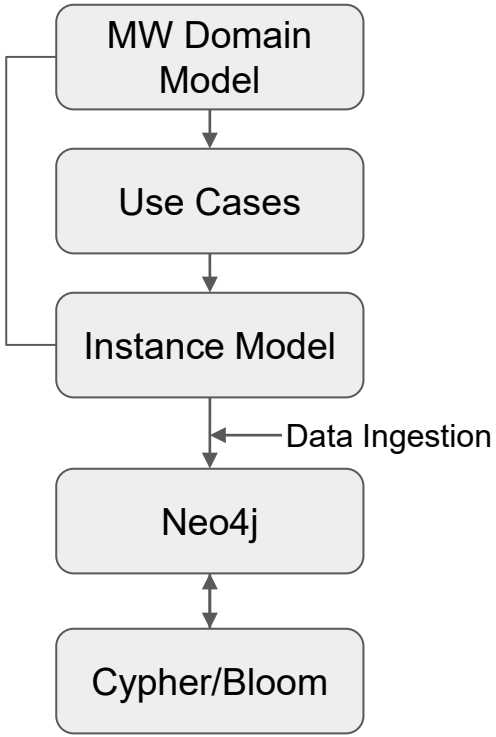
Metabolomics Workbench (MW) - Knowledge Graph (KG)

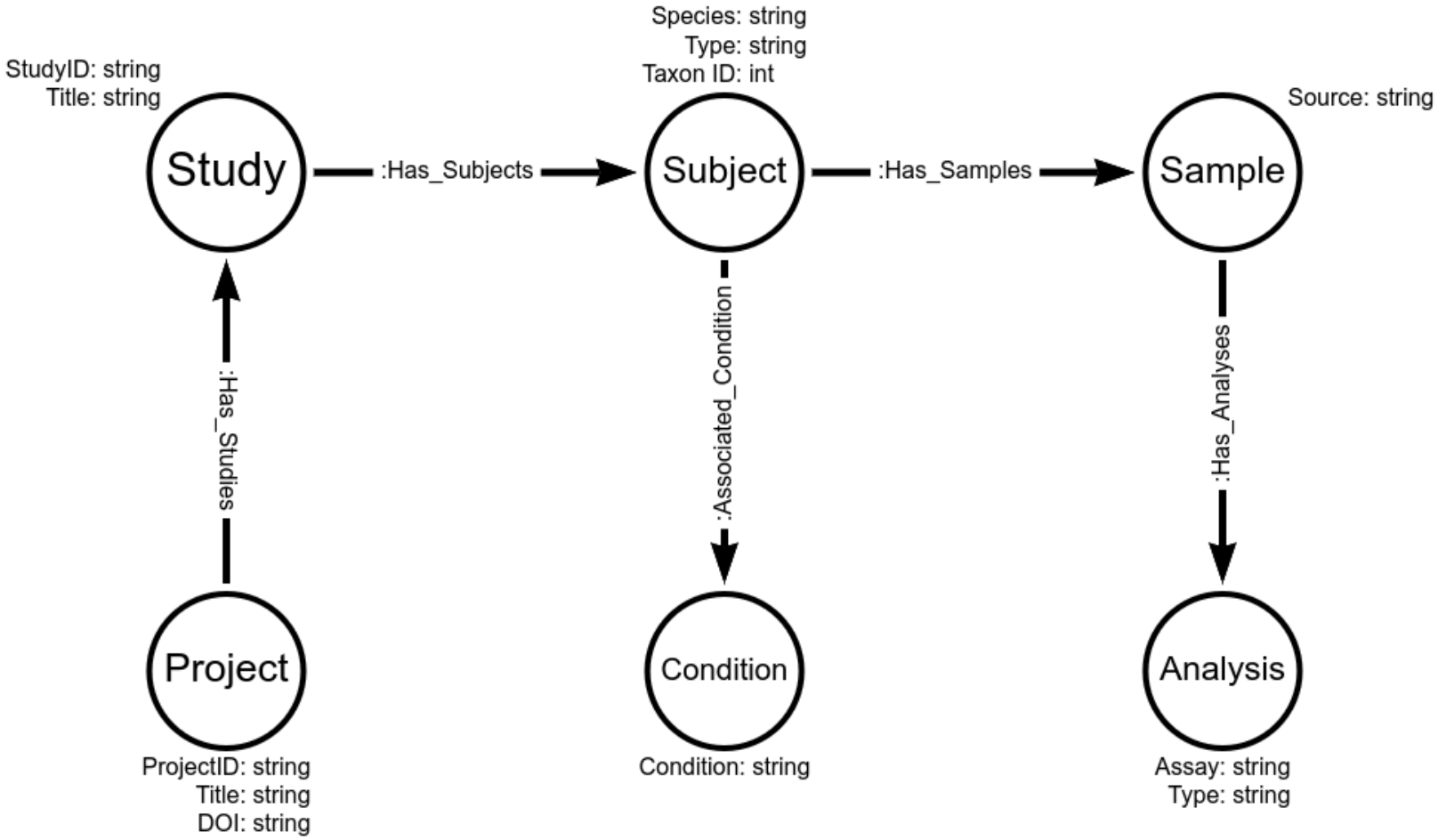
MWKG Enabled Data Search and Discovery

Graph db → Knowledge Graph

Logical Workflow



MW Domain Model



MW-KG: Use Cases

Search

- Retrieve a specific study/project and explore its data

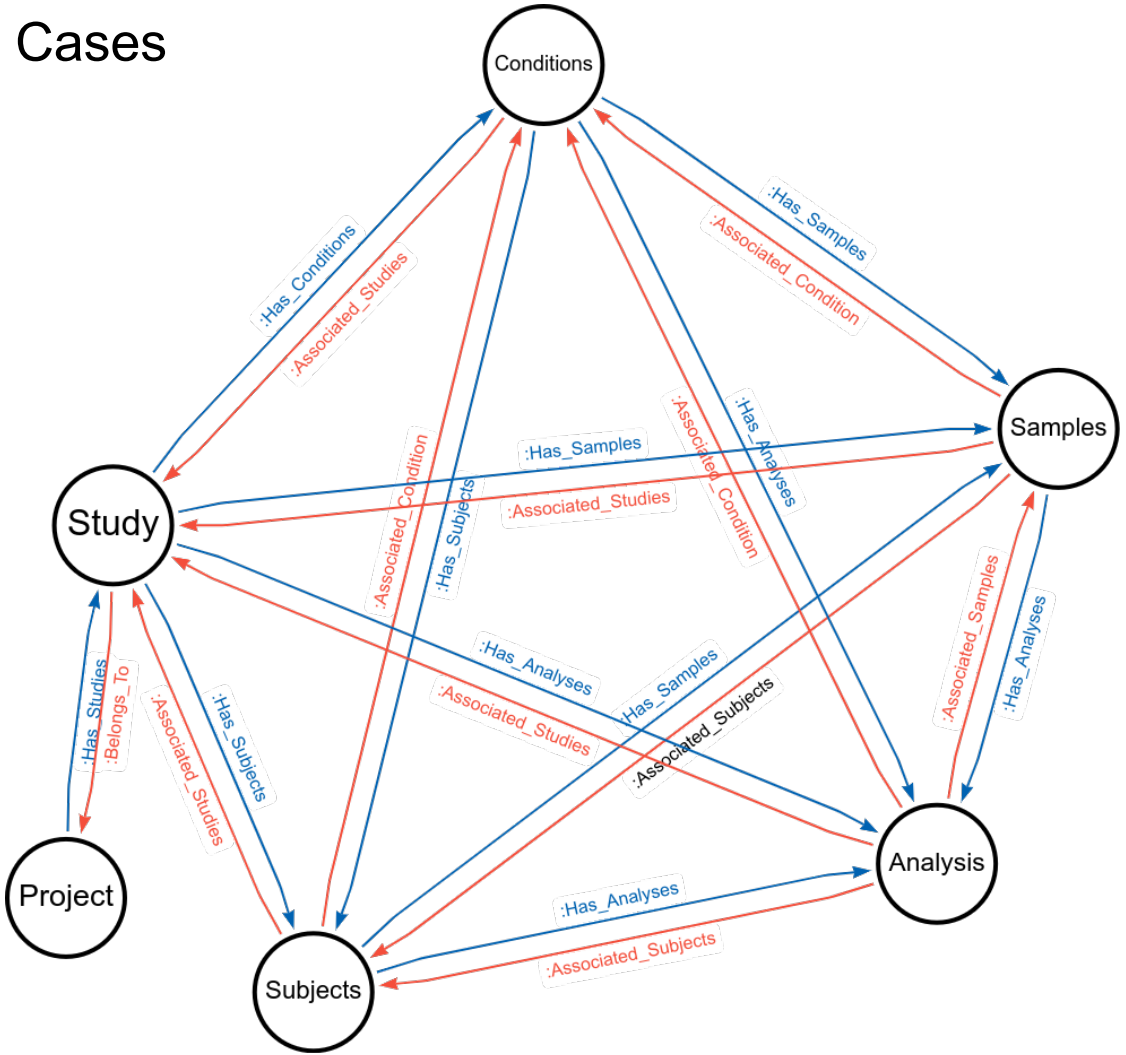
Browse → Facilitates serendipitous data discovery

- Retrieve Studies associated with a Condition (disease/phenotype/treatment)
- Retrieve Studies connected to a Sample/Species/Analytical method

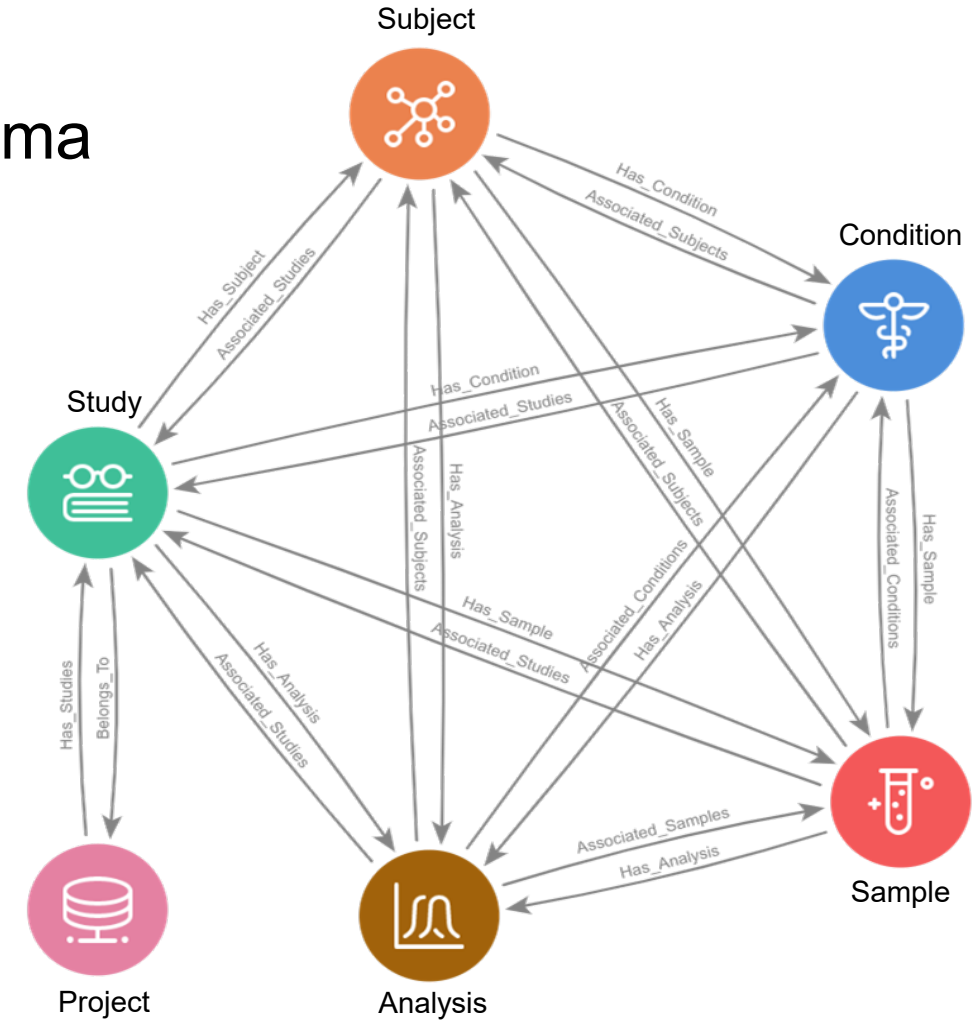
Analytics

- Centrality detection (Degree/Eigenvector/PageRank)
- Community detection (Louvain/Label propagation)
- Finding common link between entities (Shortest Path)

MW-KG Instance Model for Use Cases



MWKG db Schema



Postgres query on MW (NMDR) DB to generate the data used in neo4j DB

```
\copy (SELECT distinct st.project_id as ProjectId, pr.project_title as Title, pr.doi as DOI, st.study_id as
StudyID, st.study_title as StudyTitle, su.subject_id as subject_id, su.subject_species AS subject_species,
su.subject_type as subject_type, su.taxonomy_id AS taxonomy_id, trim(ssdm.source) as Sample,
trim(ssdm.disease) as Disease, (CASE WHEN(upper(st.analysis_type_detail) LIKE 'NMR%') THEN 'NMR'
ELSE 'Massspec' END) as Analysis,
                    st.analysis_type_detail as AnalysisType FROM
(subject su INNER JOIN
(metadata md INNER JOIN
(ssd_metabolites ssdm INNER JOIN
(project pr INNER JOIN
(study st INNER JOIN study_status_prod st_s ON st.study_id = st_s.study_id)
ON st.project_id = pr.project_id)
ON ssdm.study_id = st.study_id)
ON md.study_id = st.study_id)
ON su.subject_id = md.subject_id)
WHERE (st_s.status = 1 AND (ssdm.source !='' AND ssdm.source !='-'))
ORDER BY st.project_id, st.study_id, su.subject_id, su.subject_species, Sample)
TO project_study_species_source_for_MWKG.tsv WITH DELIMITER E'\t' NULL '' CSV HEADER;
```

1078 unique projects
1662 unique studies and subjects
134 unique species
156 unique sample sources
139 unique diseases

Data ingestion & System Resources

A cypher script was written to ingest the csv file bearing the

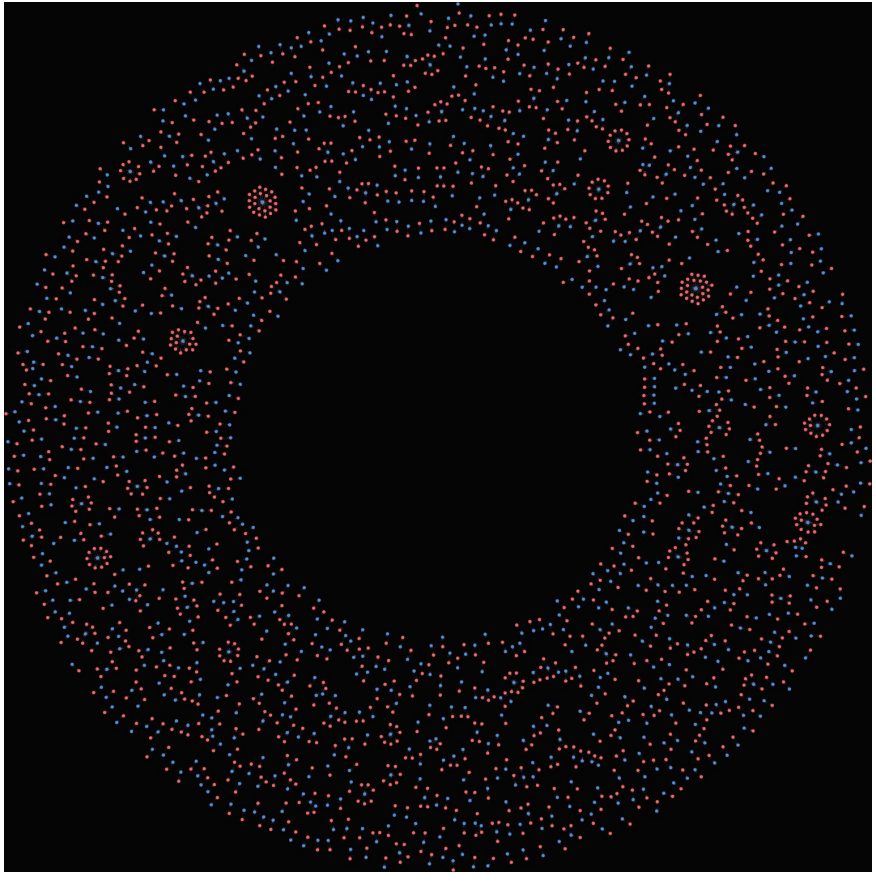
- Nodes (Project, Study, Subject, Sample, Condition, Analysis)
- Relationships (10 relationship types)
- Attributes/Properties associated with Nodes and Relationships

Into the Neo4j graph db.

A local instance of the Neo4j Desktop (ver. 1.5.8) along with Bloom (ver. 2.9.0) were installed on Ubuntu 22.04.

Cypher Query: Studies Per Project

```
MATCH p=(:Project)-[:Has_Studies]->(:Study) RETURN p
```



Studies per Project as a Table

CYPHER code to return #Studies/Project:

```
MATCH (p:Project)
WITH collect (p) AS projectbundle
UNWIND projectbundle AS projects
MATCH (projects)-[r:Has_Studies]->()
RETURN
    projects.Title AS Project_Title,
    projects.Id AS Project_Id,
    projects.DOI AS DOI,
    COUNT(r) AS Number_Of_Studies
ORDER BY Number_Of_Studies DESC
```

- The output can be exported as a .csv or .json object
- Requires knowledge of CYPHER query language

Bloom - Visualization Tool

- Bloom is a Neo4j graph data visualization tool.
- Enables users to interact with the graph db with no coding.
- Intuitive UI lets the user explore the data with the underlying instance model.
- It catalogs the node labels, their relationships, and their attributes from the graph db.
- It prompts the User to input node labels, relationships and attributes relevant to the context and enable data exploration and discovery.
- It enables, store and share Cypher queries

Use case 1: Data Search

- Retrieve the Project “PR000854”, its associated Studies, Samples, and Analysis.

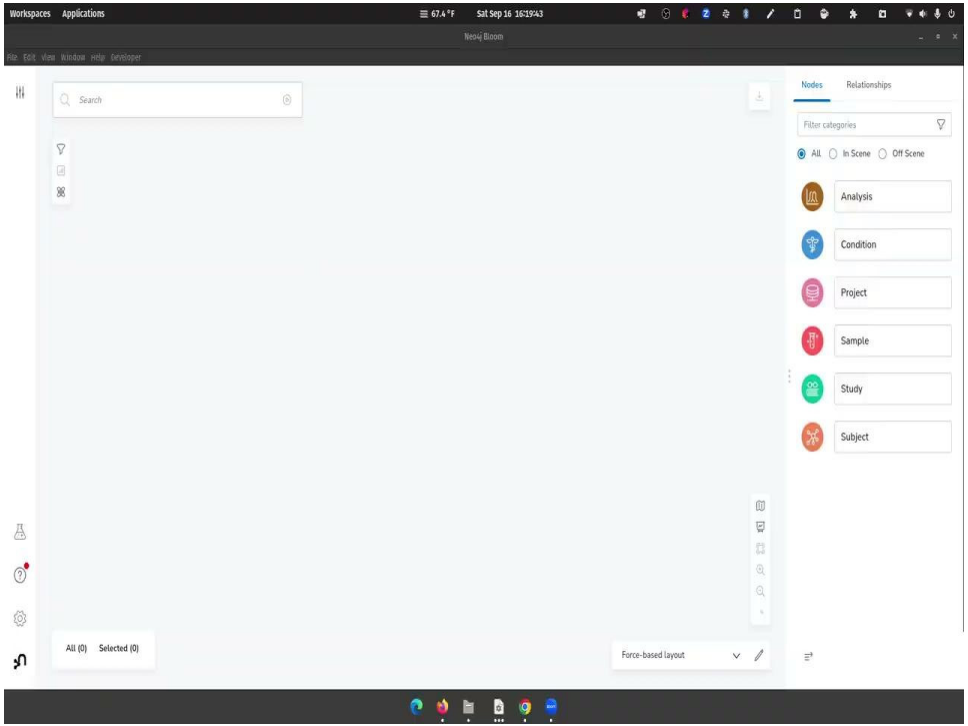
Step 1: Select ‘Project Node’ in the Search Bar. Bloom prompts to input Project Relationships, related Nodes, Attributes like ID, Title, DOI, etc., from the Graph db.

Step 2: After choosing Id, Bloom prompts the properties associated with Id, such as ‘equals’, ‘does not equal’, ‘contains’, ‘starts with’, and ‘ends with’.

Step 3: Input “PR000854” in the Id property equals. This fetches the Project Node

Step4: Expanding the Project node brings up all the nodes related with it, including the Study, Subject, Samples, Condition, and Analysis.

- Bloom fetches the Project and all its nodes, intuitively and interactively.


Workspaces
Applications
67.4 °F
Sat Sep 16 16:19:43
Neo4j Bloom
File Edit View Window Help Developer
Search
Nodes
Relationships
Filter categories
All
In Scene
Off Scene
Analysis
Condition
Project
Sample
Study
Subject
All (0)
Selected (0)
Force-based layout

Use Case 2: Data Discovery

Goal: Discover all human lung adenocarcinoma studies and associated Projects

Step 1: Input ‘Cancer’ in the Search bar → Choose (Condition) from the prompt

Step 2: Choose [:Associated_Subjects] → Choose (Subject) with {Type=Human}

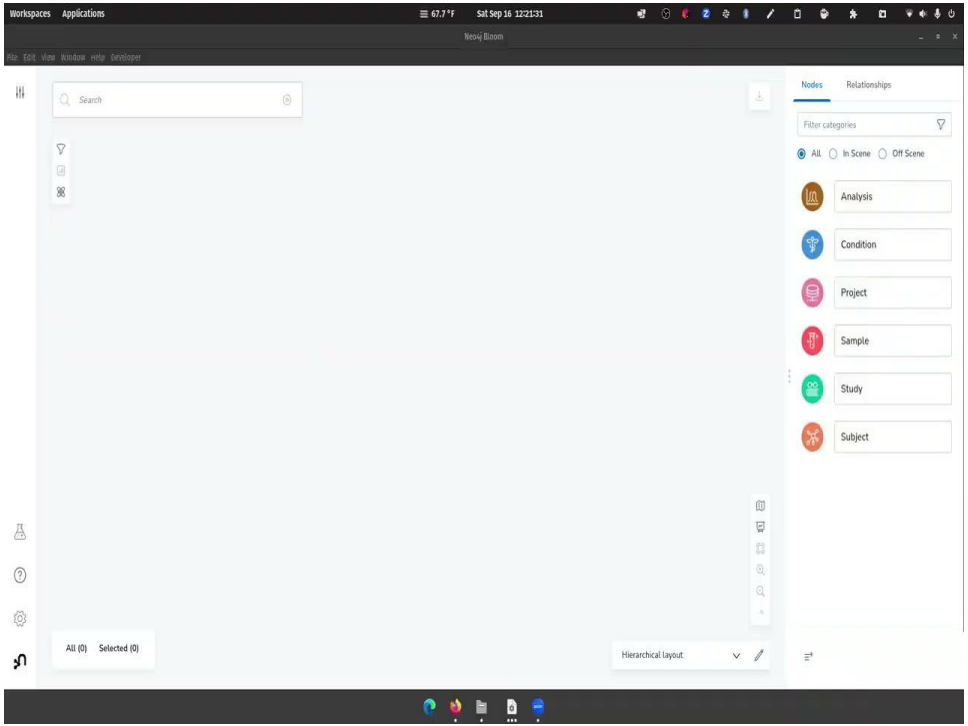
This fetches all the Studies with the Condition=Cancer AND Subject=Human

Step 3: Use Filter Option to filter the (Study) with {Title=Lung}

Step 4: Use Filter option again (Cascading filter) on (Study) with {Title=adeno}

This filters all Human Lung Adenocarcinoma studies from the MW db

There are many ways to achieve the end result. One can start with Human (Subject)→ (Condition); (Condition)→ (Subject); (Study) → Filter Lung AND Adeno → (Project) → Expand all associated nodes with the filtered Study results.


Workspaces
Applications
67.7 °F
Sat Sep 16 12:21:31
Neo4j Bloom
File Edit View Window Help Developer
Search
Nodes
Relationships
Filter categories
All
In Scene
Off Scene
Analysis
Condition
Project
Sample
Study
Subject
All (0)
Selected (0)
Hierarchical layout

Use Case 3: Analytics- Degree Centrality

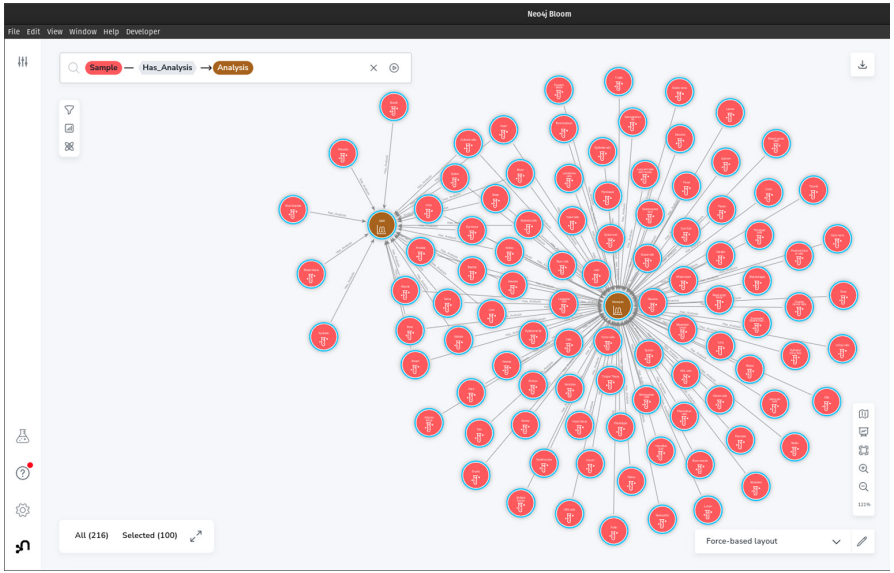
Most samples were subjected to one type of Analysis (either MS or NMR). However, some samples are subjected to both types of analysis. Identifying such samples (clusters/community) based on number of edges (some cases weights associated with such edges) is possible by applying graph analytics.

Step 1: (Samples)-[:Associated_Analysis] →(Analysis)

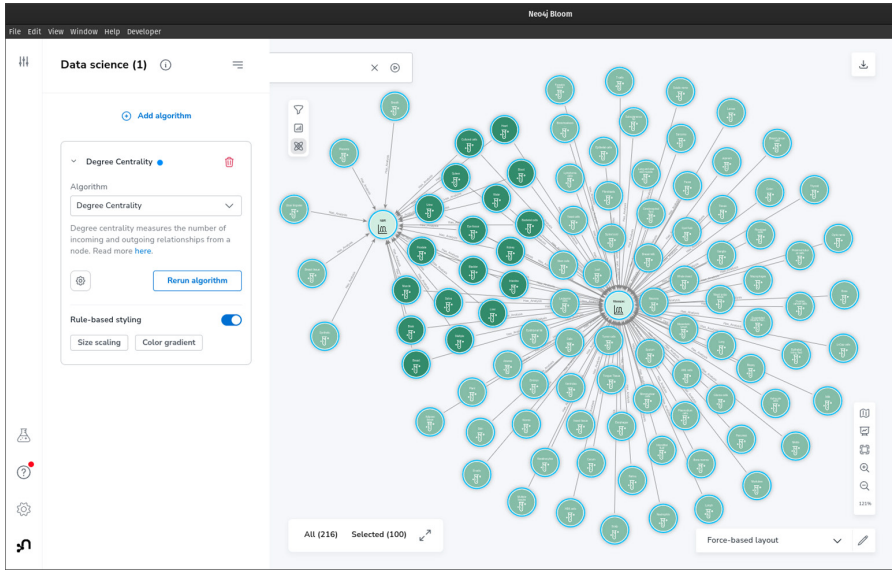
Step 2: Apply ‘Degree Centrality’ to identify the nodes with maximum number of connections (incoming and outgoing edges)

This is useful to identify such nodes when the number of nodes are large in number.

(Samples)-[:Has_Analysis] →(Analysis)



Degree Centrality identifies Samples with both Analysis



References

Metabolomics Workbench: https://metabolomicsworkbench.org

Neo4j: https://neo4j.com/

Bloom: https://neo4j.com/docs/bloom-user-guide/current/about-bloom/

Graph Data Science: https://neo4j.com/product/graph-data-science/