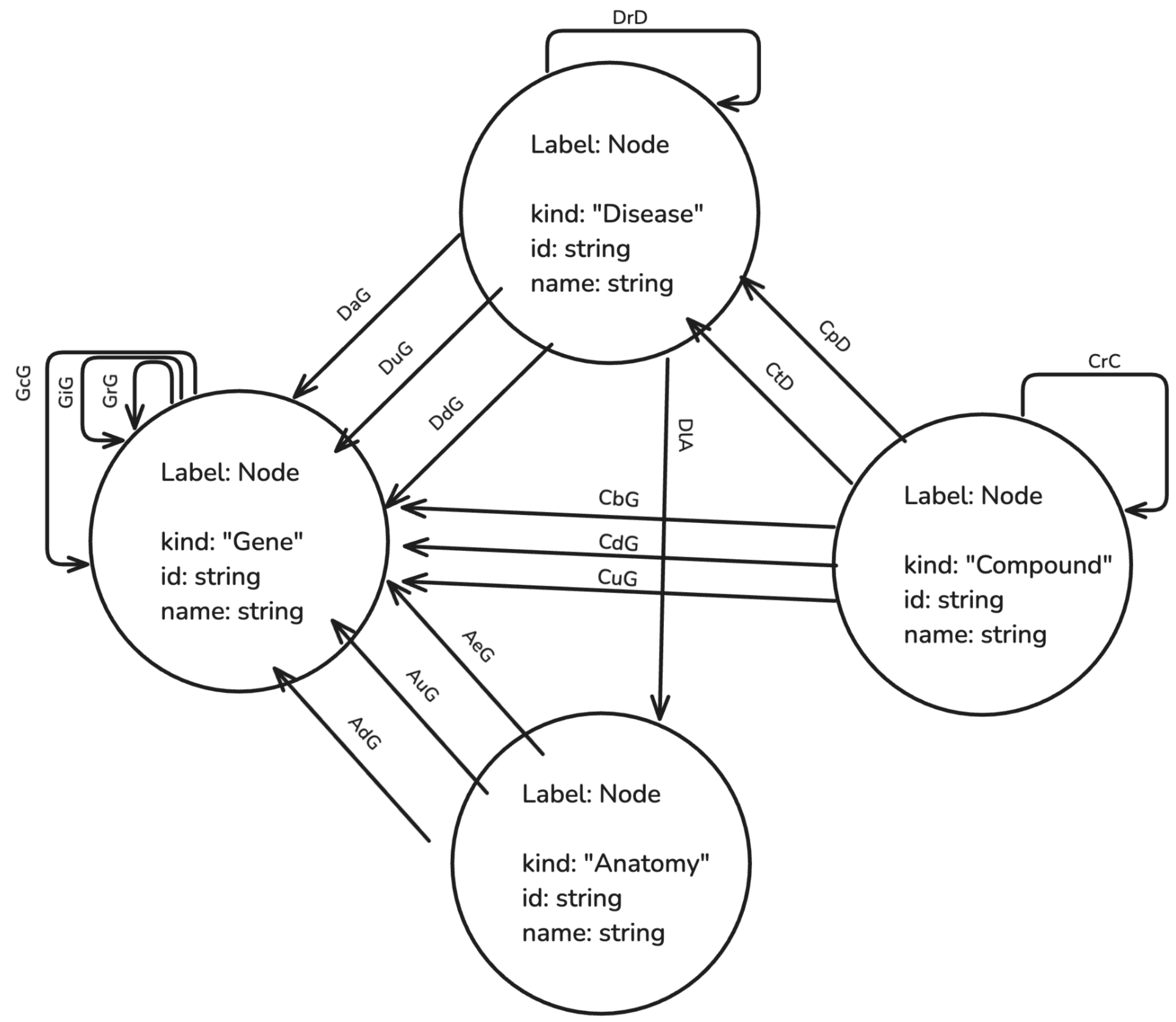
Shadman Rakib & Khandakar Wahiduzaaman
Big Data Technologies Project 1

Neo4J Design Diagram

Initially, the different nodes had different labels based on the kind. This made batching using UNWIND complicated to make performant, and the creation of the database edges took a very long time. To remedy this, we used a single node label called Node and distinguished between the types using the kind field and an index on that kind field. There was also a uniqueness constraint on id to improve performance.



Neo4J Queries

There are comments to explain the queries. They use the param $diseaseId.

Query 1 - find drugs, genes, and anatomies associated with a disease id:

```
MATCH (d:Node {kind: "Disease", id: $diseaseId})
// Match compounds that treat or palliate the disease
```

```
OPTIONAL MATCH (d)<-[r:CtD|CpD]-(c:Node {kind: "Compound"})
// Match genes associated with the disease
OPTIONAL MATCH (d)-[:DaG]->(g:Node {kind: "Gene"})
// Match anatomical locations related to the disease
OPTIONAL MATCH (d)-[:DlA]->(a:Node {kind: "Anatomy"})
RETURN d.name AS disease_name,
  collect(distinct c.name) AS compound_names,
  collect(distinct g.name) AS gene_names,
  collect(distinct a.name) AS anatomy_locations
```

Query 2 - find new potential drugs:

```
// get all compounds that can have potential to do opposite of some anatomy on a gene
MATCH (c:Node {kind: "Compound"})-[:CuG|CdG]->(g:Node {kind:
"Gene"})<-[:AdG|AuG]-(a:Node {kind: "Anatomy"})
// narrow to down to just the opposite
WHERE (c)-[:CuG]->(g)<-[:AdG]-(a)
  OR (c)-[:CdG]->(g)<-[:AuG]-(a)
MATCH (d {kind: "Disease", id: $diseaseId})-[:DlA]->(a)
// make sure the compound is a new one
WHERE NOT EXISTS ((c)-[:CtD]->(d))
RETURN DISTINCT c.name as drug_name, c.id as drug_id
```

Neo4J Potential Improvements

To make the graph more descriptive, we can consider bringing back kind labels instead of a generic Node label. However, to do this, we would have to do more research on how to more efficiently batch creation of nodes and edges.

The queries can also be improved by experimenting with what we match first compared to later. Since these different attributes have different quantities, ordering of the matching can have a great impact.

We can potentially speed up queries further by looking into more indexing techniques to help improve the performance. We can also use the EXPLAIN cypher feature to debug and understand the execution of the query to identify bottlenecks.