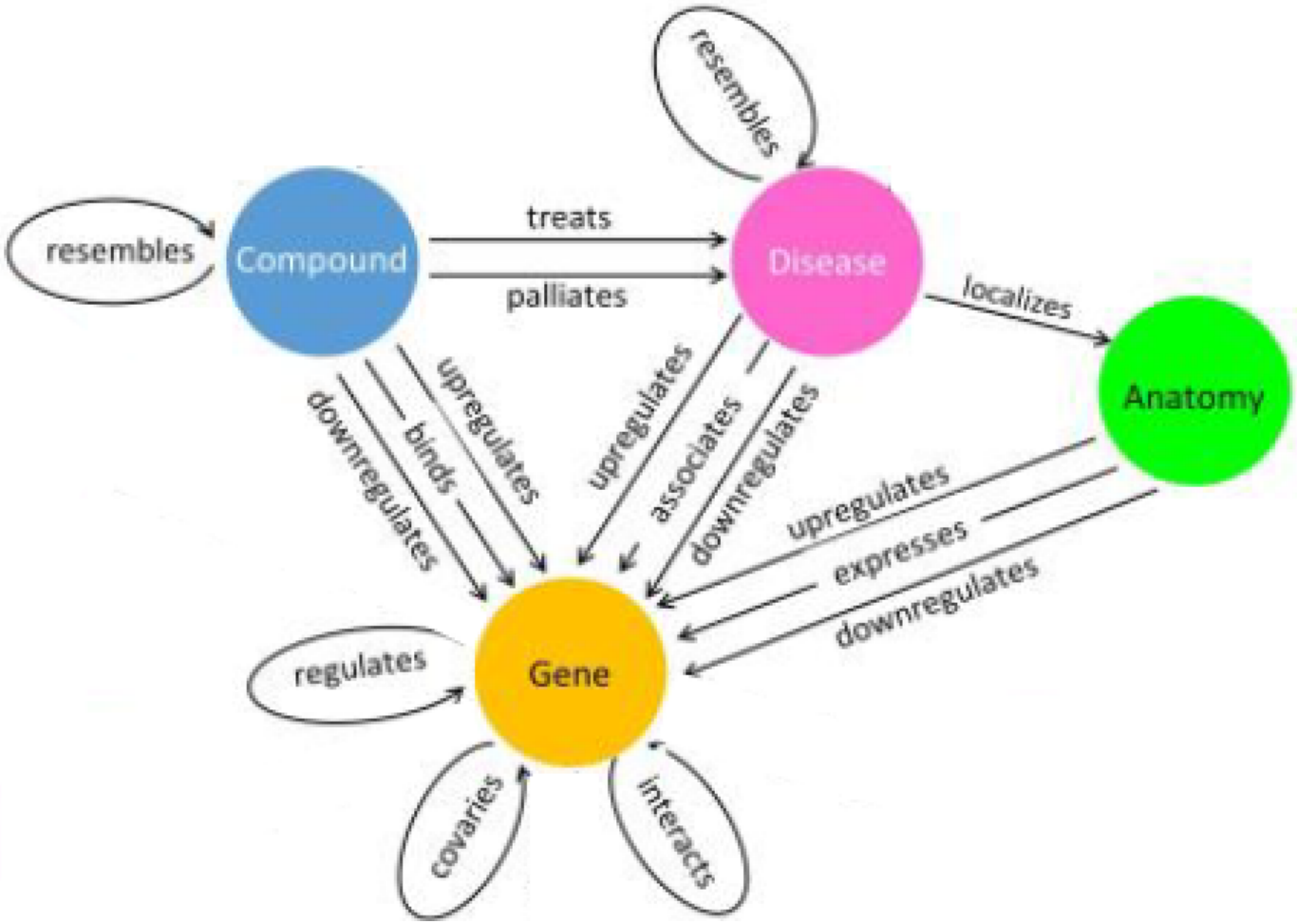
Project I: User Case

- HetioNet



Project I: User Case

- HetioNet
- Nodes

nodes.tsv

Id	name	kind
Gene::9997	SC02	Gene
Compound::DB09028	Cytisine	Compound

- Relationships

edges.tsv

source	metaedge	target
Gene::801	GiG	Gene::7428
Disease::DOID:263	DuG	Gene::857

Project I: Requirement

- Build a database to model *HetioNet*
- The database should at least answer the following questions in the quickest response time:
  - Given a disease, what is its name, what are drug names that can treat or palliate this disease, what are gene names that cause this disease, and where this disease occurs? Obtain and output this information in a single query.
  - Supposed that a drug can treat a disease if the drug or its similar drugs up-regulate/down-regulate a gene, but the location down-regulates/up-regulates the gene in an opposite direction where the disease occurs. Find all drugs that can treat new diseases (i.e. the missing edges between drug and disease). Obtain and output the drug-disease pairs in a single query.

Project I: Requirement

- A Python command-line client interface for database creation and query
- Use at least two types of NoSQL stores (Key-value, Document, Column Family, or Graph)

Project I: Requirement

- Document (no hand-writing, in print!)
  - Design diagram
  - All queries
  - Potential improvements (e.g. how to speed up query)
- All source codes (sent by email)
- Two-person team for undergraduates, work individual for MS students
- Due: 11:59pm, October 30
- Project demo: 4:30-7:30pm, October 31

Project I: Rubric

- Database design: 30%
  - Rationale (15%)
  - Implementation (15%)
- Query functionality: 40%
  - 20% each query
- Client interface: 20%
- Presentation: 10%