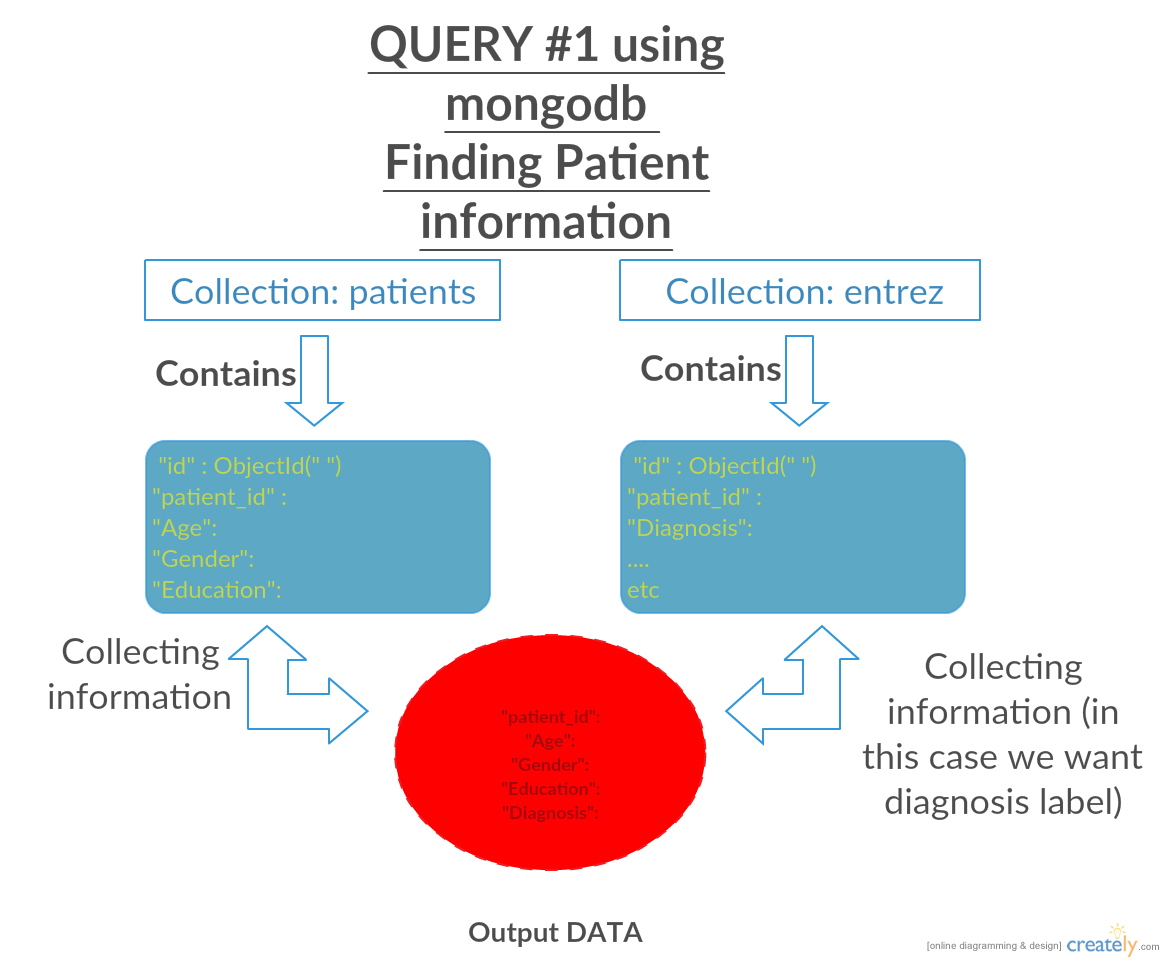
Jeong Yang & Tasdique Chowdhury

Professor Xie

Big Data

Documentation of Project 1

**Query 1: Given patient\_id, find any information for the patient**

Design: 

All queries:

USER request for patient information.

USER must correctly enter the patient\_id that is in the database.

Database will collect the data from 2 collections since patients Collection contains the information regarding the patient and entrez Collection contains the diagnosis label.

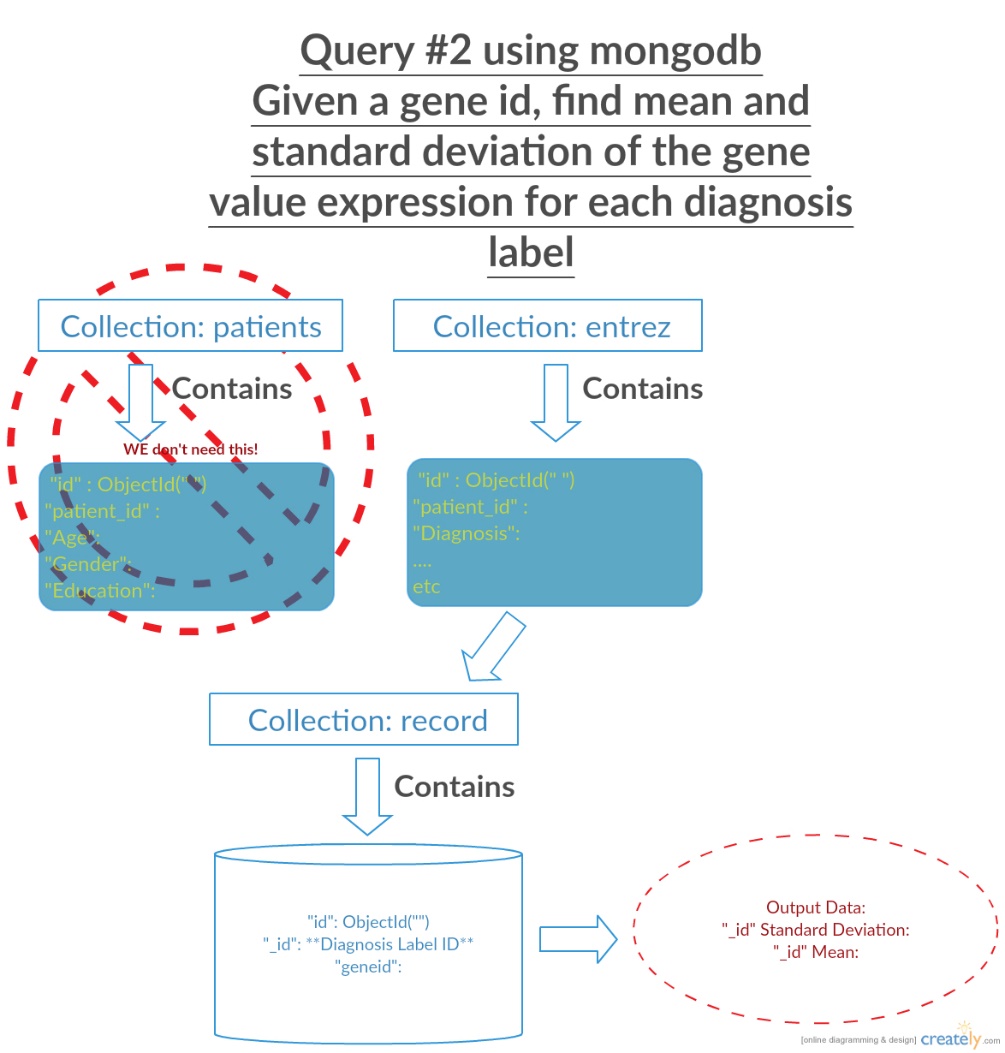
Python Terminal will give the Age, Gender, Education, and the Diagnosis Label.

**WARNING!: There is no error checking in the python code so user MUST enter the correct value for the patient\_id!**

ANY Potential improvement:

* We can grab a certain field from the entrez collection, which is the Diagnosis, and insert the Diagnosis into the patient collection so its faster to access from 1 collection rather to access from 2 collection simultaneously.

**Query 2: Given a gene, find mean and std of gene expression values for AD/MCI/NCI, respectively.**

Design:

All queries:

USER request for standard deviation and mean of the gene value expression

USER enters the geneid

Database will collect the geneid and diagnosis label and move to the record Collection to compute the data.

Python terminal will give standard deviation and mean of the gene value expression for each diagnosis label

**WARNING!: There is no error checking in this python code so USER must enter the correct value of the geneid.**

**Query 3: Given a gene, find all of its n order interacting genes.**

**WARNING: This is incomplete. Only complete through Neo4j Cypher shell. Could not complete due to time constraints.**

**The following is the Cypher code necessary to execute the necessary aspects:**

1. // CREATE to create entrez genes

USING PERIODIC COMMIT 500

LOAD CSV WITH HEADERS FROM "file:/BIOGRID-MV-Physical-3.4.144.tab2.csv" AS biogrid

CREATE(e1:Entrez {id: toInt(biogrid.`Entrez Gene Interactor A`)})

CREATE(e2:Entrez {id: toInt(biogrid.`Entrez Gene Interactor B`)})

1. // deletes excess nodes

MATCH (n:Entrez)

WITH n.id as id, collect(n) AS nodes

WHERE size(nodes) > 1

FOREACH (n in tail(nodes) | DELETE n)

1. //creates unique nodes

CREATE CONSTRAINT ON (entrez:Entrez) assert entrez.id IS UNIQUE

1. //creates relationships

USING PERIODIC COMMIT 500

LOAD CSV WITH HEADERS FROM "file:/BIOGRID-MV-Physical-3.4.144.tab2.csv" AS biogrid

MATCH (a:Entrez {id: toInt(biogrid.`Entrez Gene Interactor A`)})

WITH biogrid, a

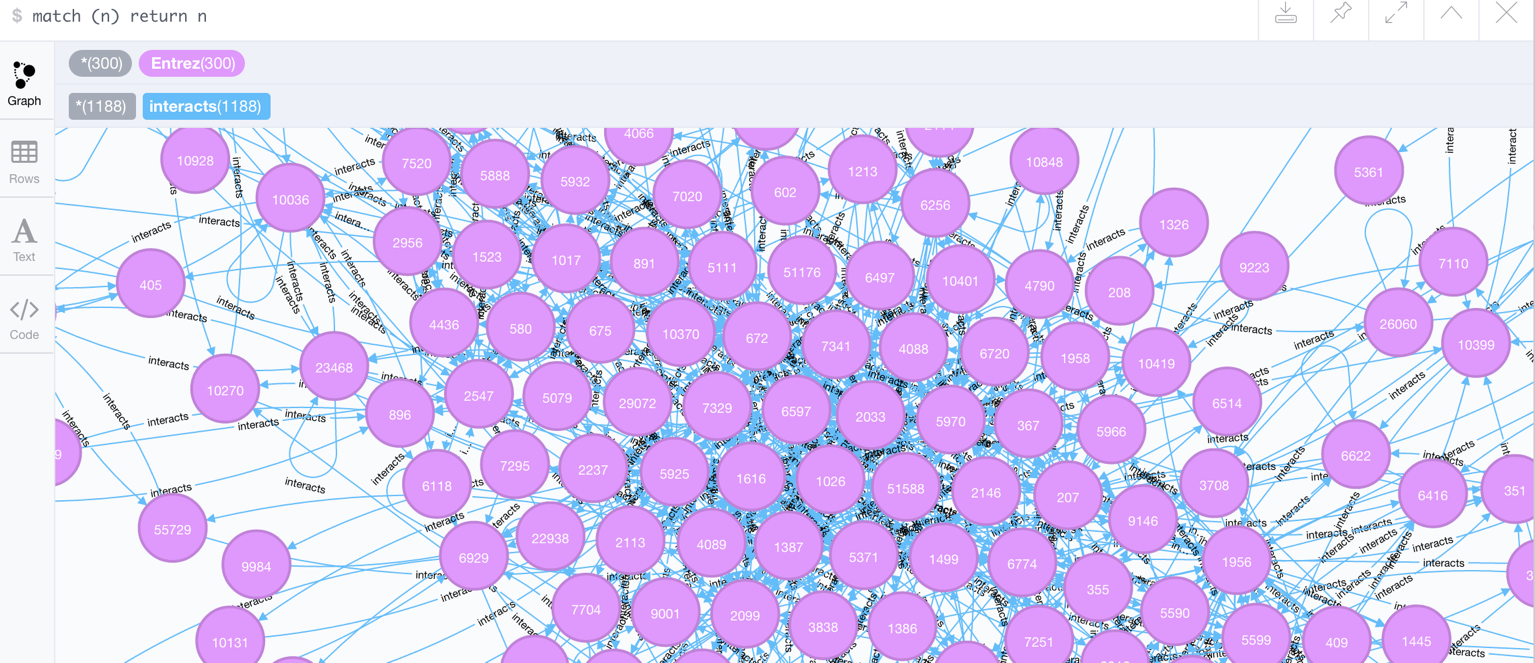
MATCH (b:Entrez {id: toInt(biogrid.`Entrez Gene Interactor B`)})

WITH a, b

MERGE (a)-[:interacts]->(b);

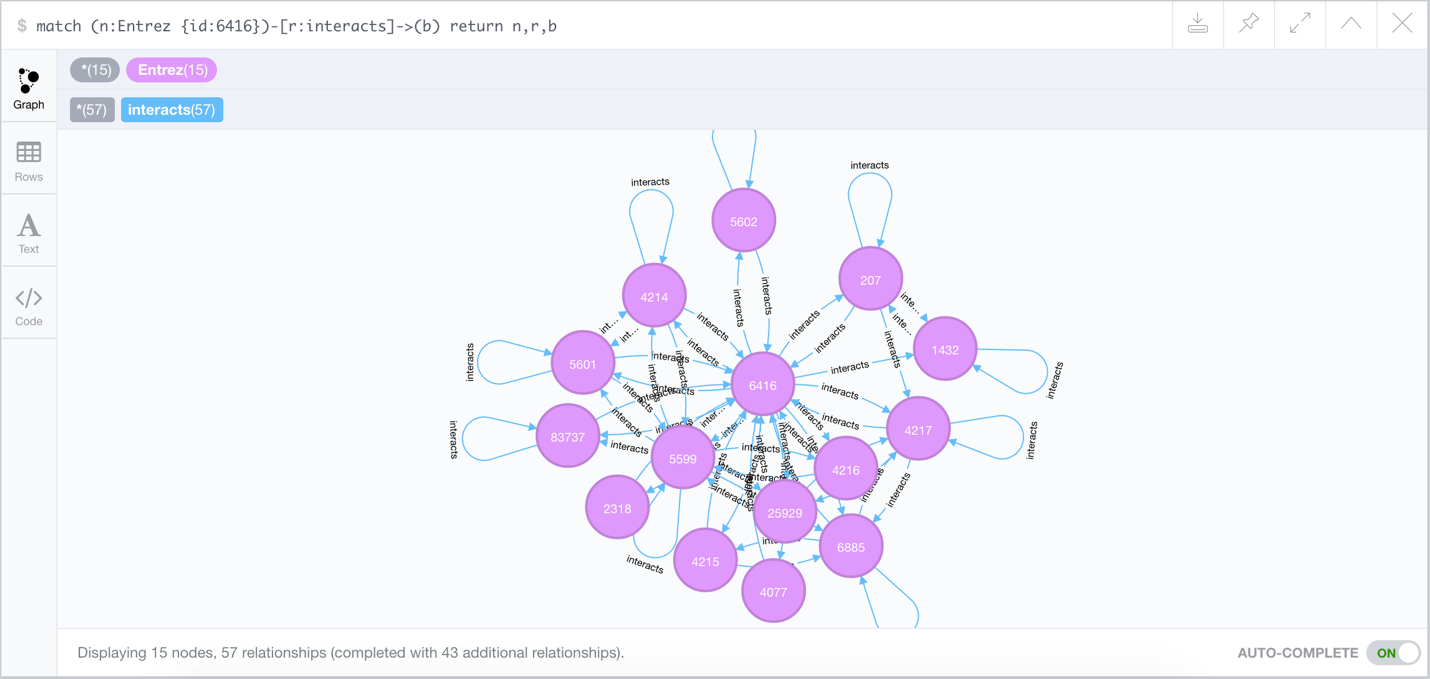
1. // shows snippet of graph

MATCH (n) return n



1. //shows subset of whole graph, with entrez id 6416

match (n:Entrez {id:6416})-[r:interacts]->(b) return n,r,b



1. // deletes all nodes and relationships to not kill application or storage memory.

MATCH (n) DETACH DELETE n

All queries (ideally):

USER requests for gene interactors.

Database gathers information from biogrid file and creates graph (using first four commented codes)

USER inputs a valid entrez id.

Database returns all n order interacting genes to entrez id inputted through Python terminal (using 6th commented code).

USER requests exit, and database runs deletion of nodes and relationships (using 7th commented code). Can be run again if requested.

ANY POTENTIAL IMPROVEMENT:

* Runs on Python.