Lab 11: Working with Graphs

In this module, we talked about both the analysis of graphs and the challenges in storing and retrieving them. In this lab, we'll use Neo4J, a Graph Database to efficiently store and retrieve graph data, and perform some interesting analysis around superheroes.

Part 1: Download and install Neo4J

Download and install neo4j community edition from http://neo4j.com/download/. Unzip the package and start neo4j. Instructions below should work for both Mac OS and EC2/Vagrant instances. **Note:** If you're running neo4j on ec2, make sure port 7474 is open in your instance's security group.

```
<navigate to the directory where neo4j was downloaded>
tar xvzf neo4j-community-2.3.0-unix.tar.gz
cd neo4j-community-2.3.0
bin/neo4j start
```

Browse to http://localhost:7474/ or http://ec2hostname:7474/ if you're running under EC2. In the browser window, connect to the database, change your password.

If you have never used neo4J before, it is recommended that you:

- Follow the "Start Learning" track
- At the end of the "Start Learning" track, choose "Jump Into Code"
- Follow the Movie Tutorial to get comfortable with Cypher

Part 2: Exploring Marvel Character Relationships

Now that you're comfortable with neo4j, we're going to use it to answer questions about the Marvel Comics Universe. First, we need to clean up our database and load the new data.

Clear the graph

```
MATCH (n)
OPTIONAL MATCH (n)-[r]-()
DELETE n,r
```

Load the nodes

```
LOAD CSV FROM
"https://s3.amazonaws.com/ucbw205data/hero_nodes.csv" AS line
MERGE (:Hero {name:line[0] , degree: toInt(line[1])})
```

Create an Index on Hero Names

```
CREATE INDEX ON :Hero(name)
```

Load the edges

```
LOAD CSV FROM
" https://s3.amazonaws.com/ucbw205data/hero_edges.csv" AS line
MATCH (u:Hero {name:line[0]})
MATCH (v:Hero {name:line[1]})
CREATE UNIQUE (u) -[:APPEARED { w: toInt(line[2])}]-> (v)
```

Part 3 Query Patterns

Finding Characters By Name

```
MATCH (spiderman:Hero)
WHERE spiderman.name STARTS WITH "SPIDER"
RETURN spiderman
```

Finding the shortest path between 2 characters

```
MATCH p=(peter:Hero {name: 'SPIDER-MAN/PETER PAR'})-
[:APPEARED*0..2]-(logan:Hero {name: 'WOLVERINE/LOGAN'})
RETURN p, length(p)
ORDER BY length(p)
LIMIT 1
```

Finding and Counting Friends of Friends

```
MATCH (peter:Hero { name: 'SPIDER-MAN/PETER PAR' })-
[:APPEARED*2..2]-(friend_of_friend)
WHERE NOT (peter)-[:APPEARED]-(friend_of_friend)
AND friend_of_friend.name <> 'SPIDER-MAN/PETER PAR'
RETURN friend_of_friend.name, COUNT(*)
ORDER BY COUNT(*) DESC , friend_of_friend.name
```

Finding Visualizing Connections Between Friends of Friends

```
MATCH (peter:Hero { name: 'SPIDER-MAN/PETER PAR' })-
[:APPEARED*2..2]-(friend_of_friend)
WHERE NOT (peter)-[:APPEARED]-(friend_of_friend)
AND friend_of_friend.name <> 'SPIDER-MAN/PETER PAR'
RETURN friend_of_friend
LIMIT 20
```

Finding Teammates

```
MATCH (tony:Hero {name:'IRON MAN/TONY STARK'}) -[e:APPEARED]->
(other) <-[f:APPEARED]- (donald:Hero {name:'THOR/DR. DONALD
BLAK'})
RETURN other
ORDER BY e.w DESC, f.w DESC
```

```
LIMIT 5
```

Part 4: Answering Questions

Modify the above templates to find answers to the following questions. Submit these answers to your instructor as the deliverable for this assignment.

1. What is the shortest path between DR. STRANGE and DR. DOOM?
2. List the 5 shortest paths between DR. STRANGE and DR. DOOM
3. List 5 Friends of Friends with the most connections and COLOSSUS II.
4. Visualize 10 Friends of friends for IRON MAN
5. Discover how the Avengers grew over time from 5 to 10. Find team members starting with 5 and incrementing to 10. Who was added to the team? Is the resulting graph ever **not** fully connected?