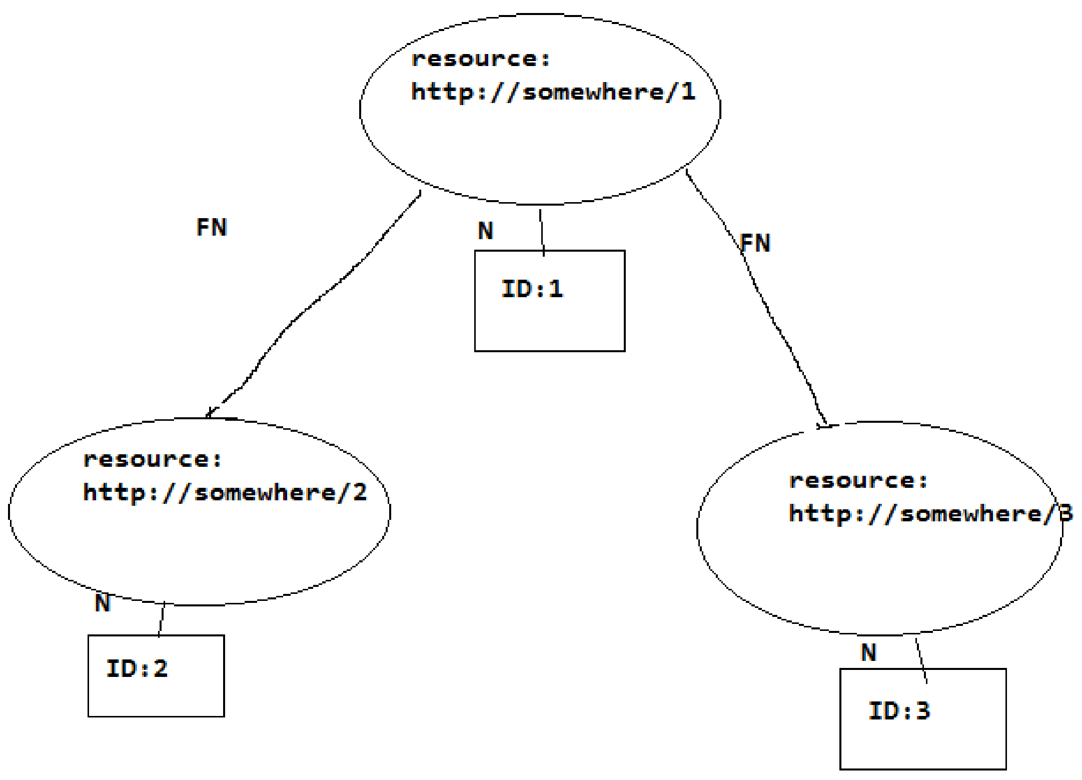
Howework2

My structure of database

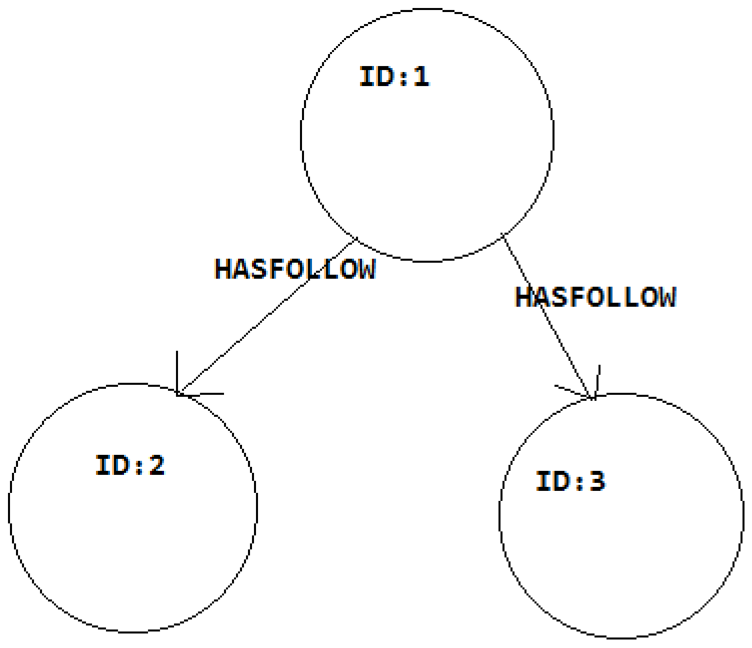
I create more than 1500 nodes into database.

Jena:



For each twitter, it has a resource and a property and I use FN representing “has follower” and use N representing “has ID”.

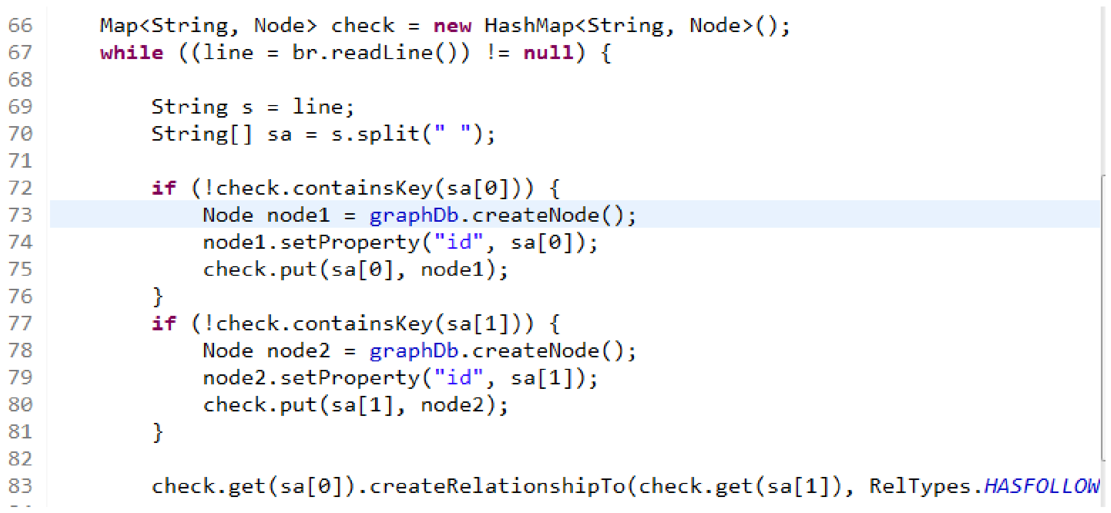
Neo4j:



Each node has a property id and I build a relationship named HASFOLLOW.

Some challenge:

Unique node is the biggest challenge I met. Compared with neo4j, Jena is more easy to use because jean is defaulted to use unique node. In neo4j, if you create a node with same property of existed node, it will create a new instance, which will result in a lot of same subject. I dealt with it by HASHMAP. Before I create a node, I will check it in a HASMAP whether it has been created. Unfortunately, in neo4j, one more null node will be created.



```
Map<String, Node> check = new HashMap<String, Node>();
while ((line = br.readLine()) != null) {

    String s = line;
    String[] sa = s.split(" ");

    if (!check.containsKey(sa[0])) {
        Node node1 = graphDb.createNode();
        node1.setProperty("id", sa[0]);
        check.put(sa[0], node1);
    }
    if (!check.containsKey(sa[1])) {
        Node node2 = graphDb.createNode();
        node2.setProperty("id", sa[1]);
        check.put(sa[1], node2);
    }

    check.get(sa[0]).createRelationshipTo(check.get(sa[1]), RelTypes.HASFOLLOW
```

HASHMAP for unique node

Performance comparisons:

I run all query in Eclipse by Java.

Neo4j seems much quicker than jena.

Query all nodes in Jena:

```
675ms 701ms 657ms 646ms 744ms
```

Query all nodes in Neo4j:

```
358ms 340ms 361ms 330ms 336ms
```

Query all followers of one node in Jena:

```
663ms 694ms 637ms 806ms 663ms
```

Query all followers of one node in Neo4j:

```
813ms 610ms 555ms 550ms 640ms
```