Traversal works differently on embedded and rest api

New issue

Closed **ml054** opened this issue Apr 16, 2015 · 1 comment

ml054 commented Apr 16, 2015

Please have a look at following failing test.

We create simple graph,
r1 -> r2 -> r3, r1->r2,
k1 -> r1, k2 -> r2, k3 -> r3, k1 -> r3

On embedded mode it returns correct results. Using REST api it doesn't work properly.

I'm using Neo4j 2.1.7 as remote server.

I'm not 100% sure that the bug is on neo4j side, it also might be caused by spring-data-neo4j. (I'm using version 3.3.0.RELEASE)

```
import static org.junit.Assert.assertEquals;

import java.util.ArrayList;
import java.util.Arrays;
import java.util.Collections;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import org.junit.Test;
import org.neo4j.graphdb.Direction;
import org.neo4j.graphdb.GraphDatabaseService;
import org.neo4j.graphdb.Node;
import org.neo4j.graphdb.RelationshipType;
import org.neo4j.graphdb.ResourceIterable;
import org.neo4j.graphdb.ResourceIterator;
import org.neo4j.graphdb.Transaction;
import org.neo4j.graphdb.factory.GraphDatabaseFactory;
import org.neo4j.graphdb.traversal.TraversalDescription;
import org.neo4j.graphdb.traversal.Traverser;
import org.springframework.data.neo4j.core.GraphDatabase;
import org.springframework.data.neo4j.rest.SpringCypherRestGraphDatabase;
import org.springframework.data.neo4j.rest.SpringRestGraphDatabase;
import org.springframework.data.neo4j.support.DelegatingGraphDatabase;

@SuppressWarnings("deprecation")
public class TestBoth {

  private enum Rels implements RelationshipType {
    RR,
    RK
  }

  @Test
  public void testEmbedded() {
    GraphDatabaseService embedded = new GraphDatabaseFactory().newEmbeddedDatabase( "db");
    GraphDatabase database = new DelegatingGraphDatabase(embedded);
    test(database);
  }

  @Test
  public void testRemoteOld(){
    GraphDatabase database = new SpringRestGraphDatabase("http://localhost:7474/db/data");
    test(database);
  }

  @Test
  public void testRemoteNew(){
    GraphDatabase database = new SpringCypherRestGraphDatabase("http://localhost:7474/db/d
```

Labels
None yet

Milestone
No milestone

Assignee
No one assigned

2 participants

```
    GraphDatabase database = new SpringCypherRestGraphDatabase("http://localhost:7474/db/d
    test(database);
  }

  public Map<String, Object> toMap(String key, String value) {
    Map<String, Object> m = new HashMap<>();
    m.put(key, value);
    return m;
  }

  public void test(GraphDatabase database) {
    try (Transaction tx =  database.beginTx()) {
      database.queryEngine().query("match ()-[r]-() delete r", null);
      database.queryEngine().query("match (n) delete n", null);
      tx.success();
    }

    Node r1, r2, r3;
    Node k1, k2, k3;

    try (Transaction tx =  database.beginTx()) {
      r1 = database.createNode(toMap("name", "r #1"), Arrays.asList("r"));
      r2 = database.createNode(toMap("name", "r #2"), Arrays.asList("r"));
      r3 = database.createNode(toMap("name", "r #3"), Arrays.asList("r"));
      database.createRelationship(r1, r2, Rels.RR, Collections.EMPTY_MAP);

      database.createRelationship(r2, r3, Rels.RR, null);
      database.createRelationship(r1, r3, Rels.RR, null);

      k1 = database.createNode(toMap("name", "k #1"), Arrays.asList("k"));
      k2 = database.createNode(toMap("name", "k #2"), Arrays.asList("k"));
      k3 = database.createNode(toMap("name", "k #3"), Arrays.asList("k"));

      database.createRelationship(k1, r1, Rels.RK, null);
      database.createRelationship(k2, r2, Rels.RK, null);
      database.createRelationship(k3, r3, Rels.RK, null);
      database.createRelationship(k1, r3, Rels.RK, null);
      tx.success();
    }

    try (Transaction tx =  database.beginTx()) {
      TraversalDescription description = database.traversalDescription()
        .breadthFirst()
        .relationships(Rels.RK, Direction.OUTGOING)
        .relationships(Rels.RR, Direction.OUTGOING);

      Traverser traverser = description.traverse(k1);
      ResourceIterable<Node> nodes = traverser.nodes();
      List<String> names = new ArrayList<>();
      try (ResourceIterator<Node> iterator = nodes.iterator()) {
        while (iterator.hasNext()) {
          Node n = iterator.next();
          names.add((String)n.getProperty("name"));
        }
      }
      assertEquals(4, names.size());
      tx.success();
    }
  }
}
```

jexp commented Apr 16, 2015 Collaborator

Yes that is to be expected.
The RestGraphDatabases have only available what the REST API offers, which is a tiny subset of the real traversal API.

If you want to fully use the traversal API and not Cypher over the wire, I recommend writing a server extension that executes your embedded code and provides the results as a custom http endpoint, you can find examples in the Manual and e.g. on Max' blog

jexp closed this Apr 16, 2015