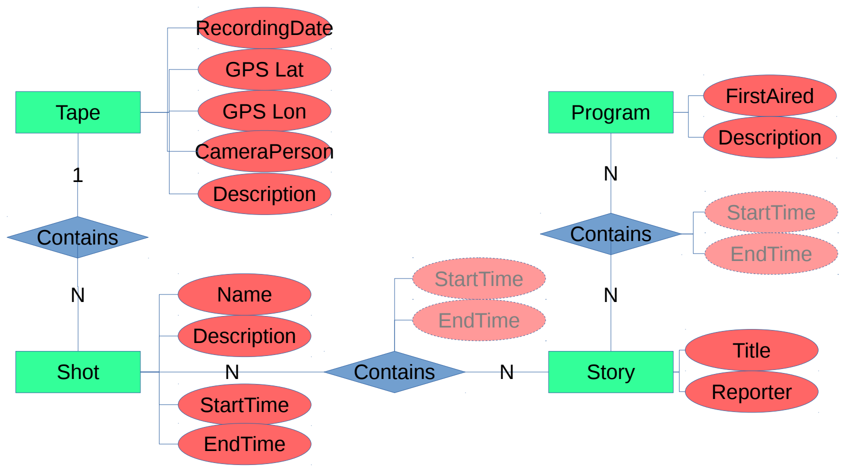
CISC 7610X, Homework 2 – Due 3/28/16

Prof Michael Mandel – mim@sci.brooklyn.cuny.edu

1 Introduction

For this assignment, we will be implementing the database from homework 1 in Java as a collection of objects and persisting them to the Neo4j graph database. You will turn in a directory of source code, an executable jar file, and a brief report describing your work.

The entity-relationship diagram for the database is:



Implementing the startTime and endTime properties on the join tables is option in this assignment, which is why they are drawn differently.

2 Setup Neo4j

2.1. Download Neo4j community edition for your platform from http://neo4j.com/download/ and unzip it.

2.2. Follow the guide on using the Neo4j browser: http://neo4j.com/developer/guide-neo4j-browser/. When prompted, **set the password for user "neo4j" to "cisc7610"** so that eventually I can use your code on my setup without recompiling it. It will guide you through the process of starting the server running locally on your computer and connecting to it through your laptop.

2.3. Make sure that you run the intro, concepts, and movie graph examples using the `:play intro`, `:play concepts`, and `:play movie graph` commands in the browser window. This is just to get familiar with Neo4j and the interface.

2.4. You might also want to read this article to get started http://neo4j.com/developer/graph-db-vs-rdbms/

2.5. After running the movie graph example, open the sidebar by clicking on the three circles in the top left of the screen. Select the entire database by clicking on the "*" button under "Node labels". Take a screenshot of the graph to be included in your report.

3 Setup Neo4j-OGM

Neo4j includes an object-graph mapping (OGM) module that will convert between java objects and Neo4j graph elements (nodes and edges). It includes three main components of interest to us:

- A set of annotations to be applied to your classes and to their fields to allow them to be persisted to the database
- A session manager to connect to the database using a URL that you supply and credentials for the database
- Various methods to query the database via the session

It is available at `https://github.com/neo4j/neo4j-ogm`.

I used version 1.1.6, as opposed to the 2.0.0 release candidate. I used it through Apache Maven, a build tool for java. If you are not familiar with Maven, this tutorial is useful: `https://maven.apache.org/guides/getting-started/maven-in-five-minutes.html`. You can add the Neo4j OGM to your project just by adding several lines to your Maven configuration file, **`pom.xml`**, described in the Neo4j OGM README: `https://github.com/neo4j/neo4j-ogm`.

4 Implement the ER diagram in Java

4.1. Implement one class for each entity in the ER diagram: **`Tape`**, **`Shot`**, **`Story`**, and **`Program`**. These classes should have a constructor that takes no arguments, but can also have a constructor that takes all of the arguments necessary to initialize them fully.

4.2. Implement relationships with other entities as a field that is a **`java.util.Set`** of the other entity class. For many-to-many relationships, it is sufficient to have one of these sets in the entity on either side of the relationship.

4.3. Implement methods to connect up related entities so that the set on both sides of each relationship is correctly populated.

4.4. Implement a **`Main`** class that instantiates the other classes and connects them together using the same data that we used in Homework 1.

5 Annotate your Java classes

5.1. Follow the tutorial on the Neo4j OGM `http://neo4j.com/docs/ogm/java/stable/#tutorial`, but instead of using the example code, apply the same concepts to your own java code.

5.2. Note that annotations live in the package **`org.neo4j.ogm.annotation`** and the session lives in the package **`org.neo4j.ogm.session`**.

5.3. As described in the tutorial, create a superclass for all of your entity classes to keep track of IDs and hashing.

6 Write your `Main` class

6.1. Continue following the tutorial and add the code concerned with the session and querying the database to your **`Main`** class.

6.2. Make sure that the Neo4j server you started running earlier is still running and have your **`Main`** class connect to it using the URL `http://neo4j:cisc7610@localhost:7474/`, assuming that you are using version 1.1.6 of the Neo4j OGM.

6.3. Before it does anything else, have your **`Main`** class clear the database by calling **`session.purgeDatabase()`** where **`session`** is an instance of **`Session`** returned by **`Neo4jSessionFactory.getInstance().getNeo4jSession()`**.

6.4. Have you **`Main`** class save the entities to the database using the **`session.save()`** function. Note that saving an object will save all of the objects linked to it, so you probably don't have to call **`save()`** on too many different objects.

7 Have the `Main` class query the database

7.1. Write code in the **`Main`** class or in the entity classes to print out a tree representation of the TV station data, with the root at a **`Program`**.

7.2. Printing a **`Program`** should print the program information followed by printing all of its constituent **`Stories`**.

7.3. Printing a **`Story`** should print its information followed by printing all of its constituent **`Shots`**.

7.4. Printing a **`Shot`** should print its information followed by the **`Tape`** id.

8 Compile and run your project

8.1. Create an executable jar file that will clear and re-populate the database when run. If using Maven, you can make the jar executable using this answer on StackOverflow: `http://stackoverflow.com/a/1814697/2037288`

8.2. Go back to the browser interface for Neo4j and again click on the "*" in the sidebar under "Node labels". It should show the graph for your project. Take a screenshot of this graph to include in your writeup.

9 Write a brief document describing your project

9.1. Include the screenshot of the movie database graph and your final graph

9.2. Include instructions for how to run your code

9.3. Describe any problems that you ran into in the course of this project

10 Submit this homework

Submit the following via the dropbox on Blackboard

- Your writeup, including the two screenshots
- The executable jar file
- A zip file containing your source code