Programming Applications with Databases

Exercise Set 8

1. Register and create a free instance in Neo4j Aura: `https://console.neo4j.io/`. Create an example Movie database based on the short script at `https://neo4j.com/docs/cypher-manual/current/clauses/match/`.
   [**1p**]

2. In the example database

   - add 2 new actors and 2 new movies,
   - add 2 new properties to 1 movie,
   - add 2 new *acted_in* relations to the existing nodes,
   - update 1 movie property,
   - remove 1 *acted_in* relation.

   [**2p**]

3. Prepare the following queries

   - return the movies where person A acted in,
   - return the movies where person A was both the actor and the director,
   - return actors who didn't play in any movie,
   - return actors who played in more than 2 movies,
   - return movies with the larger number of actors.

   In case it is needed, additional nodes or relations can be added. For more details about functions in Cypher see `https://neo4j.com/docs/cypher-manual/current/functions/`.
   [**2p**]

4. Using the code from `https://neo4j.com/docs/dotnet-manual/current/get-started/` (Example 3), implement a solution which fetches all persons from a database created in the Exercise 1 and print all details as a table in the console.
   Remark: In case some other technology is more preferred, please refer to
   https://neo4j.com/docs/create-applications/.
   [**2p**]

5. Implement a solution as specified in the previous exercise, but with the following changes:

   - all the fetched data is mapped to the appropriate object model covering both nodes and relationships, e.g. a node *Person* is mapped to the respective object of the *Person* class with all required properties,
   - extend the code to support the all CRUD operations.

   For an initial inspiration, please refer to `https://github.com/DotNet4Neo4j/Neo4jClient` (also available as a NuGet package).
   Remark: In case some other technology is more preferred, please refer to
   https://neo4j.com/docs/create-applications/.
   [**3p**]

Paweł Rajba