Databases and information systems laboratory
CS313

IIT Dharwad

Handout 8
$11 - 10 - 2023$

Consider the movie database in Neo4J. Write queries for the following.
Use the documentation from `https://guides.neo4j.com/4.0-intro-neo4j-exercises/`
If you want to look at some exercise x, go to the link:

- if $x < 10$ then use:
  `https://guides.neo4j.com/4.0-intro-neo4j-exercises/0x.html`

  Example:
  `https://guides.neo4j.com/4.0-intro-neo4j-exercises/09.html`
- if $x \geq 10$ then use:
  `https://guides.neo4j.com/4.0-intro-neo4j-exercises/x.html`

  Example:
  `https://guides.neo4j.com/4.0-intro-neo4j-exercises/12.html`

Modify nodes and relationships

1. Add a new label called *OlderMovie* for all the movies released before 2010. Check the new database schema.
2. Add a new Movie node with title *Forrest Gump*
3. For the new node added in the previous query, set the following properties:
   - released: 1994

- tagline: Life is like a chocolate Box. You never know what you gonna get
- LengthInMinutes: 142

4. Remove the *lengthInMinutes* property from the movie, Forrest Gump. Retrieve the node to confirm that the property has been removed.

5. Create the *ACTED_IN* relationship between the actors, Robin Wright, Tom Hanks, and Gary Sinise and the movie, Forrest Gump.

6. Add a new relationship called *HELPED* from *Tom Hanks* to *Gary Sinise*.

7. Add a new property called *research* to the *HELPED* relationship between *Tom Hanks* and *Gary Sinise* and set this property's value to *war history*.

8. Remove the *research* property from the *HELPED* relationship from *Tom Hanks* to *Gary Sinise*.

9. Remove the *HELPED* relationship from *Tom Hanks* to *Gary Sinise*.

10. Try to delete the movie node with title *Forrest Gump*. Did it give an error? Delete the node along with the relationships that it is involved.

11. Remove the labels *OlderMovie* and *NewMovie*

Constrainsts and Keys

1. Add a uniqueness constraint to the Person nodes in the graph where the *name* is unique.

2. Try to add a new node with name *Tom Hanks*

3. Add a constraint to say that the property *born* exists for all *Person* nodes. Does it work?

4. Ensure that the property *born* exists for all *Person* nodes, by default set it to 0. Now add the constraint in the previous question.

5. Add a new Person node, without specifying the born year. Does it work?

6. Add a constraint to say that the property *roles* exists for all *ACTED_IN* relationships.

7. Try to add a new *ACTED_IN* relationship without specifying the role. Does it work?

8. Add a constraint to say that the property *title* is unique for all *Movie* nodes.

9. Delete the constraint in the previous query.

10. Add a new constraint to the *Movie* node to assert that the title and release year together forms the key.

11. Display the list of constraints on the database.

12. Drop the constraint that requires the *ACTED_IN* relationship to have a property *roles*.

Shortest path

1. Define *Hank number* for every actor A other than Tom Hanks as follows: If A has acted with Tom Hanks in some movie then the Hank number is 0. Otherwise, Hank number of A is $i+1$ where i is the minimum among the Hank Numbers of some other actor who have acted in a common movie with A. Display the Hank number for every actor (except Tom Hanks).

Importing Data

1. Write a query to read the actor data from a file `http://data.neo4j.com/intro-neo4j/actors.csv`.

2. In the data, birth Year is a string (with a space initially), change it to integer.

3. Load the data into the graph.

Exercise

Flush the database (delete all its contents), load a new copy of the database and then try these queries. Write graph queries for the following. Submit your queries in a text file.

1. For all the movies that have been reviewed, retrieve the rating and the director(s) of the movie.

2. For every person, display the name. Further, if the person is a director then also display the list of movies that person has directed (else display `null` for this list).

3. Two actors are said to be 'co-workers' if they have acted in some common movie. Display the co-workers of *Tom Hanks* along with the title of the movie in which they have acted in common.

4. In the previous query, some actors have acted in multiple movies with *Tom Hanks* (Ex. Meg Ryan) . Modify the query such that, for every co-worker of *Tom Hanks*, display the list of movies that they have acted in common (so that every co-worker appears exactly once).

5. Retrieve pairs of all co-workers in the database. Display the pair actor names as a list along with the list of the title of the movie(s) in which both have acted.

6. In the previous query, if
   `["Hugo Weaving", "Emil Eifrem"] | ["The Matrix"]`
   is an output, then the following is also an output:
   `[ "Emil Eifrem", "Hugo Weaving"] | ["The Matrix"]`

   Modify the query to remove this redundancy (you should display only one of the two tuples in the above form)

7. For every node of the type Person, if `born` information is available. then add a new property called `Current_Age` and set its value to the current age of the person.

8. For every node of the type Person, add a new property called `Num_movies_acted` and set its value to the number of movies in which the person has acted. Set the value to 0 if the person has not acted in any movie.

9. For each person, display the name and if the person is a reviewer, display the list of movies reviewed by that person.

10. For every movie, display the number of actors acted in the movie and the number of directors of the movie.