

Applied Databases

Topic 10 Exercise Sheet

Update neo4j.conf with a new database name and type the following command:

```
tdb$ :play movies
```

This will return the following:

The screenshot shows the Neo4j Browser interface. On the left, under 'The Movie Graph', there is a 'Create' section with instructions: 'To the right is a giant code block containing a single Cypher query statement composed of multiple CREATE clauses. This will create the movie graph.' Below this, a list of steps is circled in red: '1. Click on the code block', '2. Notice it gets copied to the editor above', '3. Click the editor's play button to execute', and '4. Wait for the query to finish'. A warning message states: 'WARNING: This adds data to the current database each time it is run!'. On the right, a large code block contains a Cypher query to create a database named 'TheMatrix' with various nodes and relationships. At the bottom, a pagination bar shows '2 / 8' and navigation arrows, which is also circled in red.

Go to page 2 and follow steps 1 to 4.

A series of 171 nodes (representing Movies and People) and 253 relationships (such as ACTED_IN, DIRECTED etc.) between the nodes should now be created.

Type MATCH(n) RETURN n to see all nodes and relationships:

The screenshot shows the Neo4j Browser interface. On the left, the 'Database Information' panel is visible, showing 'Use database: tdb', 'Node Labels: (171) Movie, Person', 'Relationship Types: (253) ACTED_IN, DIRECTED, PRODUCED, REVIEWS, WRITES, FOLLOWS', and 'Property Keys: born, name, rating, released, roles, summary, tagline, title'. The main area displays a graph visualization of the movie database. The graph shows nodes representing movies and people, connected by relationships. The nodes are colored in shades of orange and green. The relationships are represented by lines connecting the nodes. The graph is titled 'The Matrix' and 'The Matrix Reloaded'. The bottom status bar indicates 'Displaying 171 nodes, 253 relationships'.

1. Write a Python programme that asks for 2 people's names from the command line. Then returns the *movie*, *released* and first 20 characters of the *tagline* properties of all movies both people have in common.
e.g.1: Both *Tom Cruise* and *Jack Nicholson* both ACTED_IN "*A Few Good Men*".
e.g.2: *Tom Hanks* ACTED_IN "*The Da Vinci Code*" and "*Apollo 13*", while *Ron Howard* DIRECTED both those movies.
2. Add the following constraint to make the *title* property of *Movie* nodes unique:
`CREATE CONSTRAINT title_unique ON (m:Movie) ASSERT m.title IS UNIQUE`

Write a Python that allows a user to enter a Movie *title*, *released* (year movie was released, and *tagline* (short description of the movie).

The movie details should then be added to the movie database.

If a movie exists with the same title in the database the user should be informed of this, otherwise the user should be informed the movie has been successfully added.