Databases and information systems laboratory
CS313

IIT Dharwad

Handout 7
04 – 10 – 2023

Install Neo4J from the following website[1]: `https://neo4j.com/try-neo4j/`.

Consider the movie database in Neo4J. Write queries for the following.

Basic retrieval

1. Retrieve the data model for the graph.
2. Query the database for all property keys.
3. Retrieve all nodes from the database. View the retrieved results as a table.
4. Retrieve all Person nodes.
5. Retrieve all movies that were released in the year 2003.
6. Retrieve all Movies released in the year 2003, returning their titles and taglines.
7. Display results for above query with headers as *Movie Title, Release Year* and *Tagline* respectively.
8. Retrieve all people who wrote the movie *Speed Racer*.
9. Retrieve all movies that are connected to the person, Tom Hanks.

```
MATCH(n:Movie)
WHERE n.title <> 'Speed Racer'
RETURN n.title
skip 3  //SKIP 3 entries
LIMIT 10 //limits the size of output
```

```
MATCH (n)
UNWIND keys(n) AS key   // UNWIND keys(n) AS key clause unwinds the keys of each node into a separate row.
RETURN DISTINCT key
```

```
MATCH(n:Person) RETURN n
```

```
MATCH(n:Movie)
WHERE n.released = 2003
RETURN n
```

```
MATCH(n:Movie)
WHERE n.released = 2003
RETURN n.title,N.tagline
```

```
MATCH(n:Movie)
WHERE n.released = 2003
RETURN n.title AS `TITLE one`,n.tagline as Tagline
```

```
Match(p:Person)-[:WROTE]->(m:Movie)
where m.title = "Speed Racer"
return m.title,p.name
```

```
Match(p:Person)-[rel]->(m:Movie)
where p.name ="Tom Hanks"
return m.title,rel,p.name
```

[1] If you are using Ubuntu and there is a problem with the installation, first run `sudo apt install libfuse2`

```
Returning Multiple Nodes:
Match(p:Person),(m:Movie)
return p,m
```

10. Retrieve information about the relationships Tom Hanks had with the set of movies retrieved earlier.
```
Match(p:Person)-[rel]->(m:Movie)
where p.name ="Tom Hanks"
return m.title,rel,type(rel),p.name        //type (rel) gives info about relationship name
```

11. Retrieve the name of the roles that Tom Hanks acted in, along with the movie title.
```
Match(p:Person)-[rel:ACTED_IN]->(m:Movie)
where p.name ="Tom Hanks"
return m.title,rel.roles
```

12. Retrieve the name of the roles of the characters in the movie The Matrix, along with the name of the actor playing the role.
```
Match(p:Person)-[rel:ACTED_IN]->(m:Movie)
where m.title="The Matrix"
return m.title,rel.roles,p.name
```

WHERE clause

1. Retrieve all movies that Tom Cruise has acted in.
```
Match(p:Person{name:"Tom Cruise"})-[rel:ACTED_IN]->(m:Movie)
return m.title,rel.roles,p.name
```

2. Retrieve all people that were born in the 70's.
```
Match(p:Person{born:1970})
return p
```

3. Retrieve the actors who acted in the movie The Matrix who were born after 1960.

4. Retrieve the names of all movie writers and the movies that they have written.
```
Match(p:Person)-[w:WROTE]->(m:Movie)
Return p.name,m.title
```

5. Retrieve all people in the graph whose *born year* data is missing.
```
Match(p:Person)
Where p.born is  NULL
Return p.name,p.born
```

6. Retrieve all movies rated by somebody. Display the movie title, name of the reviewer and the rating.
```
Match(p:Person)-[mc:REVIEWED]->(m:Movie)
Where mc.rating is not  NULL
Return p.name,m.title,mc.rating
```

7. Retrieve all actors whose name begins with James.
```
Match(p:Person)-[mc:REVIEWED]->(m:Movie)
Where p.name Starts With 'James'
Return p.name,m.title,mc.rating
```

8. Retrieve all REVIEWED relationships from the graph where the summary of the review contains the string fun, returning the movie title reviewed and the rating and summary of the relationship.
```
Match(p:Person)-[mc:REVIEWED]->(m:Movie)
Where mc.summary contains 'fun'
Return p.name,m.title,mc.summary
```

9. Retrieve all people who have produced a movie, but have not directed a movie.
```
Match(p:Person)-[pr:PRODUCED]->(m:Movie)
Where not eXISTS {Match(p:Person)-[pr:dIRECTED]->(m:Movie)}
Return p.name,m.title
```

10. Retrieve the movies and their actors where one of the actors also directed the movie.
```
MATCH (a1:Person)-[:ACTED_IN]->(m:Movie)<-[:ACTED_IN]-(a2:Person)
WHERE exists((a2)-[:DIRECTED]->(m))
RETURN a1.name as Actor, a2.name as `Actor/Director`, m.title as Movie
```

11. Retrieve all movies that were released in one of $\{2000, 2002, 2004\}$.
```
Match (m:Movie)
Where m.released in [2000, 2002, 2004]
return m.title,m.released
```

Multiple patterns

1. Retrieve all movies that *Gene Hackman* has acted in, along with the directors of the movies. In addition, retrieve the actors that acted in the same movies as *Gene Hackman*. Return the name of the movie, the name of the director, and the names of actors that worked with Gene Hackman.

```
Match (p:Person{name:"Gene Hackman"})-[a:ACTED_IN]->(m:Movie)<-[d:DIRECTED]-(p2:Person),(p3:Person)-[a2:ACTED_IN]->(m:Movie)
Return p.name,p2.name,collect(p3.name)
```

No of Hops in relationships

2. Retrieve all nodes that the person named *James Thompson* directly has the FOLLOWS relationship in either direction.

```
Match(p:Person)-[f:FOLLOWS]->(p2:Person)
Where p.name="James Thompson" or p2.name="James Thompson"
Return p.name,p2.name
```

3. Modify the previous query to retrieve nodes that are exactly three hops away.

```
MATCH (james:Person {name: 'James Thompson'})-[:FOLLOWS*3]-(other:Person)
RETURN james,other  // 3 signifies 3 hops away
```

4. Modify the previous query to retrieve nodes that are one and two hops away.

```
MATCH (james:Person {name: 'James Thompson'})-[:FOLLOWS*1..3]-(other:Person)
RETURN james,other
```

5. Modify the previous query to retrieve particular nodes that are connected, no matter how many hops are required.

```
MATCH (james:Person {name: 'James Thompson'})-[:FOLLOWS*]-(other:
Person)
RETURN james,other
```

6. Retrieve all people in the graph whose name begins with Tom and retrieve all people named Tom who directed a movie (if it exists).

```
MATCH (p:Person)-[d:
DIRECTED]->(m:Movie)
Where p.name Starts
With 'Tom'
RETURN p.name,m.title
```

7. Retrieve actors and the movies they have acted in, returning each actor's name and the list of movies they acted in.

```
MATCH (p:Person)-[d:DIRECTED]->(m:Movie)
RETURN p.name,collect(m.title)
```

8. For every movie, retrieve the title along with the number of reviewers of the movie and names of all the reviewers.

```
MATCH (p:Person)-[r:REVIEWED]->(m:Movie)
RETURN m.title, count(p),collect(p.name)
```

9. Retrieve the actors who have acted in exactly five movies, returning the name of the actor, and the list of movies for that actor.

```
MATCH (a:Person)-[:ACTED_IN]->(m:Movie)
WITH  a, count(a) AS numMovies, collect(m.title) as movies
WHERE numMovies > 1 AND numMovies < 4
RETURN a.name, numMovies, movies
```

Distinct, Min and Max

1. Retrieve the top 5 rated movies, returning the movie title and the rating.

2. Retrieve all actors that have not appeared in more than 3 movies. Return their names and list of movies.

```
MATCH (p:Person)-[a:ACTED_IN]->(m:Movie)
with p,collect(m.title) as mov_cnt
where size(mov_cnt)<=3
Return p.name,mov_cnt,size(mov_cnt)

//During Aliasing in with : Don't use attribute name
```

Lists and Date

1. Retrieve pairs of actors who have acted in the same movie. Display the pair as a list along with the list of the title(s) of the movie(s) that they have acted in common.


```
MATCH (p:Person)-[a:ACTED_IN]->(m:Movie)<-[a2:ACTED_IN]-(p2:Person)
where p2.name <> p.name
Return Distinct p.name,p2.name,collect(m.title)
```

2. For every movie, retrieve all actors that acted in that movie, and also retrieve the producers for the movie. Collect the names of the actors and the names of the producers. Return the movie titles, along with the list of actors for each movie, and the list of producers for each movie making sure there is no duplication of data. Order the results returned based upon the size of the list of actors.

```
MATCH (p:Person)-[a:ACTED_IN]->(m:Movie)<-[pr:PRODUCED]-(p2:Person)
return Distinct m.title, collect(p.name) AS ACTORS,collect(p2.name) as PEODUCERS
```

3. Modify the previous query and unwind the list of movies and then return the name of the actor and the title of the associated movie.

4. Retrieves all movies that Tom Hanks acted in, returning the title of the movie, the year the movie was released, the number of years ago that the movie was released, and the age of Tom when the movie was released.

```
MATCH (p:Person{name:'Tom Hanks'})-[a:ACTED_IN]->(m:Movie)
Return m.title,m.released,2023-m.released as `Years ago Released ` ,m.released-p.born as `Tom Hanks Age`
```

Exercise Queries

Write queries for the following. Submit them in a text file.

1. Retrieve all Movie nodes.

2. Display title, released, and tagline values for every Movie node in the graph.

3. Retrieve names of all persons in the database and display their name and born year as *Name* and *Year of birth* respectively.

4. Retrieve all Persons that are connected to the movie, The Matrix.

5. Retrieve information about the relationships of the movie The Matrix with the set of persons retrieved in the previous query.

6. Retrieve all movies released in the year 2000.

7. Retrieve all actors whose name ends with *Thompson*. Display the name of the actors and the movies they have acted in, along with the name of the roles.

8. Retrieve the movies that have the movie's title in actor's role. Return the movie title, actor name and the role.

9. Return all people who have worked along with *Tom Hanks* in some movie (either as actor or director)

10. Retrieve all movie titles in the graph and retrieve all people who have reviewed the movie along with the review summary and rating (if it exists).

11. Retrieve directors and the movies they have directed, returning each director's name and the list of movies they have directed.

12. Retrieve all movies that Tom Cruise has acted in and the co-actors that acted in the same movie, returning the movie title and the list of co-actors that Tom Cruise worked with.

13. Retrieve all directors, their movies, and people who acted in the movies, returning the name of the director, the number of actors the director has worked with, and the list of actors.

14. Retrieve the movies that have at least 2 directors, and optionally the names of people who reviewed the movies. Display movie title and the names of the reviewers.

15. Retrieve all the movies during the 1990s and return the released date, the movie title, and the collected actor names for the movie.

16. For the previous query, collect all the movies that are released in the same year (Also display the actors who have acted in some move that year).

17. Modify the previous query to order the results returned so that the more recent years are displayed first.

18. Retrieve all the actors and the list of movies where the actors have acted in 5 or more movies.