Data Engineering 1: Lab 07 - Solution

Task 01: Installation of neo4j

Install the Desktop Version of neo4j on your machine or a virtual machine. Download it from here: https://neo4j.com/download-neo4j-now

Task 02: First steps with neo4j

Start the movie graph sample on your neo4j installation. Go through the tutorial and get to know neo4j https://neo4j.com/developer/cypher/guide-cypher-basics/

Task 03: Working with neo4j

a) Create a *Movie* node for the movie with the *title*, *Forrest Gump*

```
CREATE (:Movie {title: 'Forrest Gump'})
```

b) Retrieve the newly-created node

```
MATCH (m:Movie)
WHERE m.title = 'Forrest Gump'
RETURN m
```

c) Create a *Person* node for the person with the *name*, *Robin Wright*.

```
CREATE (:Person {name: 'Robin Wright'})
```

d) Retrieve the *Person* node you just created by its *name*

```
MATCH (p:Person)
WHERE p.name = 'Robin Wright'
RETURN p
```

e) Add the label *OlderMovie* to any *Movie* node that was released before 2010.

```
MATCH (m:Movie)
WHERE m.released < 2010
SET m:OlderMovie
RETURN DISTINCT labels(m)
```

f) Retrieve all older movie nodes to test that the label was indeed added to these nodes.

```
MATCH (m:OlderMovie)
RETURN m.title, m.released
```

g) Add the label *Female* to all *Person* nodes that has a person whose name starts with *Robin*.

```
MATCH (p:Person)
WHERE p.name STARTS WITH 'Robin'
SET p:Female
```

h) Retrieve all *Female* nodes

```
MATCH (p:Female)
RETURN p.name
```

i) We've decided to not use the Female label. Remove the Female label from the nodes that have this label.

```
MATCH (p:Female)
REMOVE p:Female
```

j) Add the following properties to the movie, Forrest Gump:
released: 1994
tagline: Life is like a box of chocolates,Ä¶you never know what you're gonna get.
lengthInMinutes: 142

```
MATCH (m:Movie)
WHERE m.title = 'Forrest Gump'
SET m:OlderMovie,
m.released = 1994,
m.tagline = "Life is like a box of chocolates...you never
know what you're gonna get.",
m.lengthInMinutes = 142
```

k) Retrieve the top 5 ratings and their associated movies, returning the movie title and the rating

```
MATCH (:Person)-[r:REVIEWED]->(m:Movie)
RETURN  m.title AS movie, r.rating AS rating
ORDER BY r.rating DESC LIMIT 5
```

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

```
MATCH (a:Person)-[:ACTED_IN]->(m:Movie)
WITH  a,  count(a) AS numMovies, collect(m.title) AS movies
WHERE numMovies <= 3
RETURN a.name, movies
```

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

```
MATCH (p1:Person)-[:FOLLOWS]-(p2:Person)
WHERE p1.name = 'James Thompson'
RETURN p1, p2
```

n) Modify the query of m) to retrieve nodes that are exactly three hops away.

```
MATCH (p1:Person)-[:FOLLOWS*3]-(p2:Person)
WHERE p1.name = 'James Thompson'
RETURN p1, p2
```

o) Modify the query of m) to retrieve nodes that are one and two hops away.

```
MATCH (p1:Person)-[:FOLLOWS*1..2]-(p2:Person)
WHERE p1.name = 'James Thompson'
RETURN p1, p2
```

p) 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(m) AS numMovies, collect(m.title) AS movies
WHERE numMovies = 5
RETURN a.name, movies
```

q) 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.

```
MATCH (p:Person)-[:ACTED_IN]->(m:Movie)<-[:ACTED_IN]-
(p2:Person)
WHERE p.name ='Tom Cruise'
RETURN m.title as movie, collect(p2.name) AS `co-actors`
```

r) Retrieve the movies that have at least 2 directors, and optionally the names of people who reviewed the movies.

```
MATCH (m:Movie)
WITH m, size((:Person)-[:DIRECTED]->(m)) AS directors
WHERE directors >= 2
OPTIONAL MATCH (p:Person)-[:REVIEWED]->(m)
RETURN  m.title, p.name
```