Lab 4 Graph-oriented Databases

Objetivos

Os objetivos deste trabalho são:

- Compreender os fundamentos das bases de dados orientadas à representação de grafos.
- Instalar e utilizar uma solução de código aberto.
- Desenvolver soluções para diversos casos de uso.

Nota prévia

Este módulo deverá ser preferencialmente desenvolvido em Linux. Caso pretenda usar Windows verifique as notas sobre compatibilidade do software que irá usar.
Submeta o código/resultados/relatórios no elearning.
Bom trabalho!

4.1 Neo4j – Instalação e exploração

O Neo4j é uma base de dados de grafos desenvolvida pela Neo4j, Inc. A versão *Community Edition* é disponibilizada sob licença *GNU General Public License (GPL) v3*.

a) Instale a versão CE do Neo4j no seu computador pessoal. Siga as instruções disponíveis no sítio https://neo4j.com/download/.

b) Uma vez instalada, execute a aplicação *Neo4j Desktop*, crie uma base de dados (e.g. *testes41*) e abra o *url* disponibilizado (http://localhost:7474/browser/).

c) Estude o funcionamento do sistema, explorando os exemplos, nomeadamente a coleção *Movie*.

   Consulte os slides disponibilizados para a disciplina e sítios web com documentação sobre Neo4j:

   - *https://neo4j.com/developer/get-started/*
   - *https://neo4j.com/docs/developer-manual/current/cypher/functions/*
   - *https://neo4j.com/graph-databases-book/*

   Deve estudar conceitos e funcionalidades tais como:

   - Criação de projetos e base de dados
   - Criação de nós e de relações
   - Escrita, Leitura, Edição, Remoção (CRUD)
   - Pesquisa através da linguagem *Cypher*.

Nota: Não é necessário entregar qualquer resultado desta alínea. Contudo, recomenda-se que dispenda o tempo necessário para a compreensão dos conceitos principais, antes de avançar para as próximas alíneas.

4.2 Exploração do grafo de filmes

Na aplicação *Neo4j Desktop* crie o projeto *cbd* e dentro a base de dados *filmes*. Arranque *filmes* e importe a base de dados de filmes que já vem disponível no neo4j usando o comando:

```
$ :play movie-graph
```

Nota: depois deste processo não insira mais vértices/edges nesta base de dados. Caso pretenda fazer testes de inserção use outra BD.

Execute e analise os resultados das expressões seguintes, em *Cypher*:

```
MATCH (n) RETURN n
```

```
MATCH (n)
WITH COUNT(n) AS numVertices
MATCH (a)-[e]->(b)
RETURN numVertices, COUNT(e) AS numEdges
```

Analise e execute a expressão seguinte, que apresenta os tipos de nó que existem na base de dados:

```
match (n)
return labels(n) as labels, keys(n) as keys, count(*) as total
order by total desc;
```

Nesta expressão:

- `match ()` seleciona qualquer nó
- `match (n)` seleciona qualquer nó e atribui-o à variável `n`
- a função `labels(n)` devolve uma lista com o tipo (*label*) de cada nó `n`
- a função `keys(n)` retorna a lista de propriedades de cada nó `n`
- a função `count(n)` é uma função de agregação, neste caso conta cada grupo *labels*, *keys* (o `GROUP BY` é implícito em *Cypher*)

Da mesma forma, podemos extrair informação sobre as relações existentes:

```
match (m)-[r]->(n)
return labels(m), type(r), labels(n), count(*) as total
order by total desc;
```

Podemos ainda obter as propriedades de cada relação:

```
match ()-[r]->()
return type(r) as type, keys(r) as keys, count(*) as total
order by type;
```

Construa agora expressões *Cypher* para responder às seguintes perguntas:

Nota: Escreva todas as respostas no ficheiro `CBD_L42_<NMEC>.TXT`, onde `<NMEC>` deve ser substituído pelo seu nº mecanográfico. Para cada pergunta, N, escreva sempre uma linha anterior com o conteúdo #N. Se pretender incluir comentários use "//". Exemplo:

```
// NMEC: 12345
#1
match ...
```

1. Encontre todos os atores que dirigiram um filme em que também atuaram e apresente o nome do ator e o título do filme.

2. Para cada filme realizado depois de 2005, apresente os nomes de todos os atores que atuaram nesse filme.
3. Encontre pares de nós com mais do que uma relação entre si.
4. Encontre todos os pares de pessoas que fizeram revisões do mesmo filme. Apresente os seus nomes e título de cada filme.
5. Encontre todos os pares de atores que atuaram em vários filmes juntos.
6. Determine a idade média do elenco do filme "Apollo 13" no ano do lançamento do filme.
7. Encontre os 10 filmes com o elenco mais velho no momento do lançamento do filme. Apresente o filme e a idade média arredondada a 2 casas decimais, por ordem decrescente.
8. Apresente o subgrafo `ACTED_IN` do filme com o elenco mais novo, no momento do lançamento do filme.
9. Qual é o caminho mais curto (usando qualquer tipo de relação) entre John Cusack e Demi Moore?
10. Qual a dimensão caminho mais curto (usando qualquer tipo de relação) entre Keanu Reeves e Tom Cruise?
11. Quais são a dimensão do caminho mais curto entre pessoas com nome Jim e pessoas com nome Kevin?
12. Que pessoas têm uma distância 2 para Jim Cash (a distância entre duas pessoas é o comprimento do caminho mais curto entre eles)?
13. Qual é a maior distância de uma pessoa para Kevin Bacon?
14. Qual é a maior distância entre duas pessoas?
15. Qual é a distribuição de distâncias em pares (isto é, para a distância 1, 2, 3, ..., quantos pares de pessoas têm essa distância um do outro)?
16. Indique as 10 pessoas com menor distância média em que o caminho entre elas são relações do tipo ACTED_IN.

4.3 Análise de uma rede de programadores e de projetos

Neste exercício, pretende-se analisar uma rede constituída por projetos e seus membros, que constam de um repositório de software.
Nota: Escreva todas as respostas no ficheiro `CBD_L43_<NMEC>.TXT`

a) Tendo por base o conteúdo do ficheiro *git_selection.csv*, modele as entidades e relações necessárias para carregar o seu conteúdo no net4j.

b) Crie a base de dados *github*. Copie o ficheiro para a pasta *import* desta base de dados. Consulte o link *http://neo4j.com/docs/operations-manual/current/configuration/file-locations/* para perceber em que área é armazenada a base de dados no seu SO. Na interface *http* do Neo4j, carregue o conteúdo do ficheiro usando o comando LOAD CSV (tenha em atenção a sintaxe deste comando, a estrutura do ficheiro CSV e a modelação que fez previamente).

c) Construa expressões *Cypher* para fornecer a seguinte informação:

1. Liste a informação de cada utilizador.
2. Liste o nome de cada utilizador.
3. Liste a informação de cada projeto, no qual tenha participado pelo menos um utilizador.
4. Liste os utilizadores e total de projetos em que cada um colabora.
5. Liste os utilizadores e do total de projetos em que cada colabora ordenados por ordem decrescente do total.
6. Liste projetos e total de membros em cada projeto.
7. Liste projetos e total de membros com a role "Committer" em cada projeto.
8. Liste todos os utilizadores que participaram nos mesmo projetos do que o utilizador "atm" (id). Mostre os atributos: nome de "atm", nome utilizador 2, nome do projeto.
9. Liste todos os utilizadores que participaram com a role "Committer" em projetos em que o utilizador "atm" (id1) participou com a role "PMC".

4.4 Temática Livre - Driver (Java)

a) Instale um driver de Neo4j para Java (ou outra linguagem) e crie um pequeno programa para ligação ao servidor Neo4j, a uma nova base de dados (e.g. *lab44*).

b) Identifique um *dataset* à sua escolha, distinto dos anteriores com uma dimensão mínima de 500 nós. Pode pesquisa e descarregar da internet ou construir um, num formato que seja adequado para carregar através do driver.

c) Crie um programa para inserir o *dataset* na base de dados do Neo4j. Programaticamente, construa um mínimo de 10 queries à sua escolha e apresente no ficheiro `CBD_L44c_output.txt` cada pesquisa e o seu resultado.