Neo4J dans la pratique (TP2)

1. Préalable Neo4J

1.1 Version Neo4J autre que celle déposée sur moodle

Avant de démarrer le serveur, il faut modifier le fichier neo4j.conf (dans répertoire conf) et ajouter deux instructions (si vous ne travaillez pas avec la version Neo4J déposée sur moodle).

```
# ajout pour import/export RDF
dbms.unmanaged_extension_classes=semantics.extension=/rdf

# ajout pour autoriser export dans un fichier
apoc.export.file.enabled=true
```

Listing 1 – ajouts dans neo4j.conf

Il faut également disposer les plugins adaptés pour les procédures APOC et le plugin NeoSemantics. Pour la version 3.5.21, il s'agit des archives :

```
apoc-3.5.0.14-all.jar
neosemantics-3.5.0.4.jar
```

Listing 2 – archives utiles

1.2 Mise en route

Vous démarrerez ensuite le serveur :

```
... ./bin/neo4j start
```

Listing 3 – ordre de mise en route du serveur

2. Enrichissement du modèle

Les derniers maires successifs de Montpellier sont à ajouter au modèle

Créer les objets Personne en relation avec la commune de Montpellier (déjà présente dans la base) suivants

```
MATCH (c:Commune {name:'MONTPELLIER'})
CREATE (gf:Personne {nom:"FRECHE",prenom:"GEORGES"}) <-[ap1:ADMINISTREE_PAR
    {date_debut:1997, date_fin:2004}]- (c),
```

```
(hm:Personne {nom:"MANDROUX",prenom:"HELENE"})<-[ap2:ADMINISTREE_PAR {date_debut:2004,
    date_fin:2014}]-(c), (ps:Personne
    {nom:"SAUREL",prenom:"PHILIPPE"})<-[ap3:ADMINISTREE_PAR {date_debut:2014,
    date_fin:2020}]-(c), (md:Personne
    {nom:"DELAFOSSE",prenom:"MICKAEL"})<-[ap4:ADMINISTREE_PAR {date_debut:2020,
    date_fin:2026}]-(c)
return *
```

Listing 4 – Les maires de Montpellier

2.1 Exercices après enrichissement

1. définir l'ordre Cypher qui permet d'ajouter un label Maire aux objets de label Personne, qui sont associés à un objet de label Commune, au travers d'une relation de type ADMINISTREE_PAR
2. lister l'ensemble des procédures rendues disponibles grâce à l'ajout d'archives Java dans le répertoire plugins. Lister aussi l'ensemble des fonctions.
3. utiliser une de ces procédures du paquetage d'extension général (préfixe apoc) pour export une partie des objets du graphe au format json (apoc.export.json.query). Vous renverrez l'identifiant, les labels et le nom de la commune, ainsi que le non de son département et de sa région. Quels sont les autres formats disponibles ?

2.1.1 Utilisation du plugin Neosemantics

Neosemantics permet d'exploiter Neo4J à la manière d'un triplestore. Nous en explorons quelques fonctionnalités élémentaires.

1. vous renverrez le modèle de connaissances de la base au format RDF. Vous pouvez dessiner sous forme de graphe une partie du résultat. Que remarquez vous comme différence avec ce que vous savez contenir le modèle ?
2. Vous renverrez au format RDF, la description du noeud correspondant à la commune de MONTPELLIER à partir de son identifiant interne. Vous pouvez dessiner sous forme de graphe une partie du résultat.
3. Renvoyez le résultat de la requête suivante au format RDF

```
MATCH (c:Commune {name:'MONTPELLIER'}) RETURN c
```

Listing 5 – Requête

   Est ce que ce résultat est équivalent au résultat de la description du noeud de la commune de MONTPELLIER ?
4. Renvoyez les informations sur MONTPELLIER et ses différents maires au format RDF
5. Enrichissez l'ordre précédent pour renvoyer le plus d'informations possibles sur MONTPELLIER
6. Comment faire pour renvoyer les informations qui correspondent aux propriétés valuées de la relation ADMINISTREE_PAR au format RDF ?

3. Import de données au format RDF

Il est possible d'importer au sein d'une base de données Neo4J, des triplets provenant de points d'accès SPARQL. Ici une requête SPARQL exploitant une des procédures du plugin neosemantics (semantics.importRDF) vous est donnée. Cette requête de type CONSTRUCT exploite le point d'accès SPARQL de Wikidata pour retourner différentes informations sur des communes (ici le concept City est emprunté au vocabulaire schema.org). Auparavant, il faudra créer l'index :

```
CREATE INDEX ON :Resource(uri)
```

Listing 6 – Requête CYPHER/SPARQL

```
WITH ' PREFIX sch: <http://schema.org/>
CONSTRUCT{ ?item a sch:City;
         sch:address ?inseeCode;
         sch:name ?itemLabel ;
         sch:geoTouches ?otherItem .
      ?otherItem a sch:City;
      sch:name ?otheritemLabel ;
      sch:address ?otherinseeCode . }
WHERE { ?item wdt:P374 ?inseeCode .
   ?item wdt:P47 ?otherItem .
   ?otherItem wdt:P374 ?otherinseeCode .
      ?item rdfs:label ?itemLabel .
        filter(lang(?itemLabel) = "fr") .
      ?otherItem rdfs:label ?otheritemLabel .
        filter(lang(?otheritemLabel) = "fr") .
    FILTER regex(?inseeCode, "^34") .
      } limit 400 ' AS sparql CALL semantics.importRDF(
  "https://query.wikidata.org/sparql?query=" +
     apoc.text.urlencode(sparql),"JSON-LD",
   { headerParams: { Accept: "application/ld+json"} })
YIELD terminationStatus, triplesLoaded,namespaces,extraInfo
RETURN terminationStatus, triplesLoaded,namespaces,extraInfo
```

Listing 7 – Requête CYPHER/SPARQL

Une fois cet import terminé, vous répondrez à une série de questions.

3.1 Questions d'appropriation

1. Expliquer en langage naturel ce que renvoie la requête SPARQL construite. Faîtes un graphe rapide sur 2 cités du résultat de l'import dans Neo4J.
2. Les nœuds importés correspondent à des villes et villages de l'Hérault. Vous ferez en sorte de les lier au nœud correspondant au département de l'Hérault via la relation WITHIN
3. Vous supprimerez les nœuds de type commune du graphe
4. Renvoyez le nombre de communes limitrophes de la commune de Montpellier

3.2 Questions sur les chemins

1. renvoyez un des plus courts chemins entre Montpellier et Beaulieu
2. renvoyez tous les plus courts chemins entre Montpellier et Beaulieu
3. renvoyez le nom des cités traversées par un des plus courts chemins entre Montpellier et Beaulieu

4. renvoyez le nom et l'adresse des cités traversées par un des plus courts chemins entre Montpellier et Beaulieu
5. renvoyez le nombre des plus courts chemins entre Montpellier et Beaulieu
6. renvoyez tous les chemins entre Montpellier et Beaulieu (très coûteuse). Que faire pour réduire la complexité ?
7. retourner un des plus courts chemins qui ne passe pas par Clapiers ?