Bases de Données Spécialisées

Projet : implémentation d'une BD graphe dans Neo4j

Le projet est à faire en binôme. Vous soumettrez une archive ou un lien vers vos fichiers sources (e.g. csv, requêtes Cypher et SQL), ainsi qu'un rapport en anglais ou français au format pdf décrivant votre travail. La date limite pour soumettre ces documents sur moodle est le 26 décembre. Si vous voulez disposer d'un peu plus de temps, c'est possible, mais j'appliquerai une petite pénalité sur la note que vous obtiendrez. Je vous donnerai probablement également cette note très tard (un phénomène d'absorption par le vortex du second semestre aura vraisemblablement lieu). Un rapide oral en présentiel ou distanciel - sous forme de questions de ma part - pourra être organisé début janvier.
Vous êtes encouragés à discuter entre vous et à me poser des questions sur moodle. En revanche si vous communiquez entre groupes différents, veillez à ne pas me rendre des travaux trop similaires (utilisez en particulier des jeux de données différents).
Il est également possible de proposer un autre sujet ou de demander à travailler sur un problème plus théorique (dans ce cas, contactez moi par mail amelie@irif.fr).

1. Choix et import d'un jeu de données

Vous choisirez un jeu de données dans l'open data (vous pouvez par exemple en sélectionner un sur https://www.kaggle.com/datasets) ou bien vous récupérerez des données directement sur le web (par exemple en utilisant les librairies de la Python Data Science Stack). Veillez à ce qu'il s'agisse de données *connectées*, pour lesquelles une implémentation graphe est pertinente. Vous implémenterez vos données dans une base de données relationnelle PostgreSQL, ainsi que dans une base de données graphe Neo4j. N'oubliez pas de définir soigneusement vos contraintes d'intégrité et index (dont éventuellement des index full-text de type lucene), ainsi que d'expliciter votre stratégie d'import de données dans le rapport, qui devra également expliquer vos choix de modélisation (relisez bien les conseils de modélisation donnés en cours, il n'est d'ailleurs pas interdit de s'y référer dans le rapport !). Si vous avez du nettoyer vos données, expliquez également comment vous avez procédé.

2. Requêtes

Vous proposerez une dizaine de requêtes Cypher interrogeant votre base de données Neo4j. J'attends au moins :

- une requête avec un filtre négatif sur certains types de patterns (inexistence d'un certain pattern, voir par exemple la requête 2.2.2 du tp2)

- une requête avec `OPTIONAL MATCH`
- une requête utilisant des quantified graph patterns qu'il vous semblerait difficile d'écrire sans cette fonctionalité
- deux utilisations de `WITH` différentes (par exemple, pour filtrer les résultats d'un agrégat, pour séparer lecture et mise à jour du graphe, voir `https://neo4j.com/docs/cypher-manual/current/clauses/with/`)
- une requête explorant à la fois les données et la topologie du graphe (voir par exemple la requête 2.2.1 du tp2 retournant tous les vols avec escale assurés par air france)
- une requête utilisant `COLLECT` et `UNWIND`
- une requête manipulant des listes avec `reduce`, cf `https://neo4j.com/docs/cypher-manual/current/functions/list/`, voir page exemple slide 170 des transparents sur Cypher (Cypher.pdf, cours 2 sur moodle), lisez l'article "Dangers of List Processing in Querying Property Graphs"déposé sur moodle et testez des requêtes du type de celles qui y sont étudiées
- un filtre post `UNION` avec `CALL`
- une requête utilisant des fonctions de prédicat du type `all()`, `any()`, `exists()`, `none()`, `single()`.

Incluez dans votre rapport quelques plans d'exécution, avec et sans index. Qu'en concluez-vous ? Comparez en particulier l'évaluation de requêtes Cypher a priori équivalentes mais utilisant des syntaxes différentes (d'autant plus intéressant si l'une s'évalue bien et l'autre non). Pour chaque requête Cypher, essayez de proposez une requête SQL équivalente : vous comparerez l'efficacité de l'évaluation des deux requêtes par postgres et neo4j. En particulier, proposez au moins une requête SQL récursive sur votre BD relationnelle, dont vous comparerez l'efficacité avec celle d'une requête Cypher équivalente sur votre BD graphe. Essayez également de trouver une requête SQL plus efficace qu'une requête Cypher, à expressivité équivalente. Essayez d'expliquer vos résultats.

3. Analytique de graphe

Faites tourner quelques algorithmes de la Graph Data Science Library. Expliquez vos choix (pourquoi tel algorithme est-il particulièrement adapté à vos données ?). Utilisez des projections nommées. Réutilisez-en au moins une plusieurs fois. Essayez plusieurs modes (pas seulement stream). Utilisez éventuellement bloom pour visualiser l'analyse de vos données. Quelles connaissances pouvez-vous inférer de vos données à partir de cette analyse ?
Vous pouvez par ailleurs consulter `https://homepages.cwi.nl/~boncz/edbt2022.pdf` et comparer une requête Cypher retournant un plus court chemin version Cypher (mots clefs `shortestpath` ou `allshortestpaths`) et une requête SQL équivalente (à base de fenêtrage

et récursivité voir slides 33 à 36). Vous pouvez également comparer Dijkstra pondéré version procédure GDS appelée dans une requête Cypher sur un graphe neo4j et version SQL utilisant WITH recursive et window functions dans une requête SQL exécutée sur une base de données postgres (slides 37 à 38).

4. Partie libre

Vous pouvez choisir d'inclure à votre projet tout autre aspect que vous jugerez pertinent (e.g., implémentation alternative avec Apache TinkerPop et comparaison de l'expressivité de Gremlin et Cypher, analyse de l'expressivité de vos requêtes dans des langages théoriques de type CRPQ et extensions, import de données RDF via neosemantics, intégration de votre base de données Neo4j dans une application au moyen du driver Neo4j de votre langage de programmation préféré, écriture de trigger, de procédures user-defined, comparaison de Spark et GDS, etc...). Le nouveau driver python a l'air en particulier prometteur dans le contexte de l'utilisation du pluggin GDS de Neo4j (cf `https://neo4j.com/developer-blog/get-started-with-neo4j-gds-python-client/`).
Aucune prise d'initiative de ce type ne vous pénalisera, bien au contraire. N'hésitez pas à consulter les ouvrages recommandés sur moodle et / ou d'autres. Soyez curieux, posez moi des questions, discutez entre vous et apprenez-moi des choses. Au moins un cinquième de la note concernera cette partie libre.