Rapport du TP2 : Exploration des Projections Natives et Orientation des Relations dans Neo4j GDS

DONE BY :

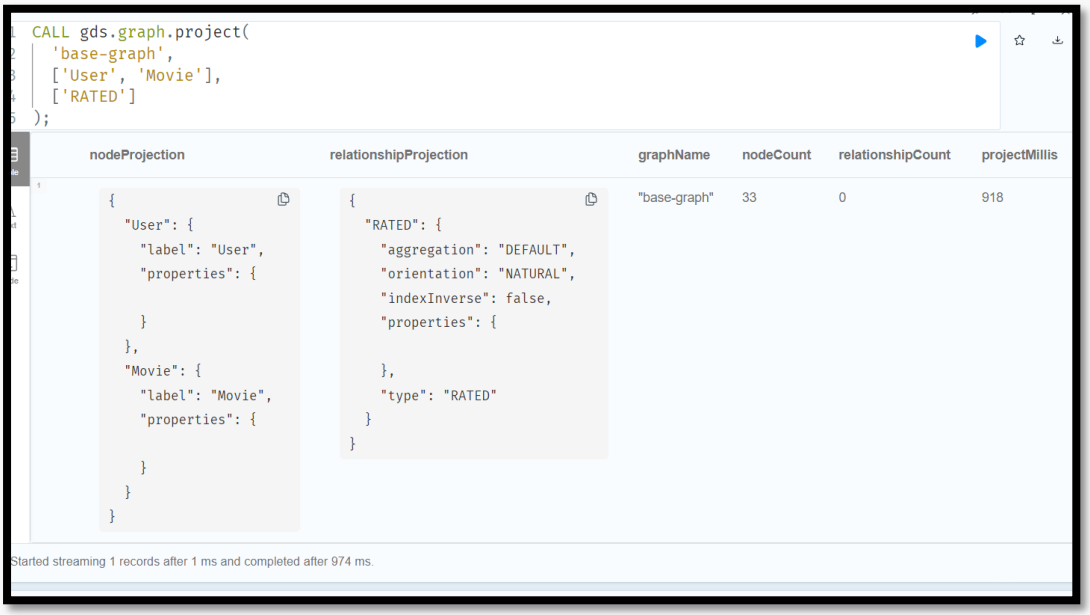
- **IMANE MALIKI**

Objectif :

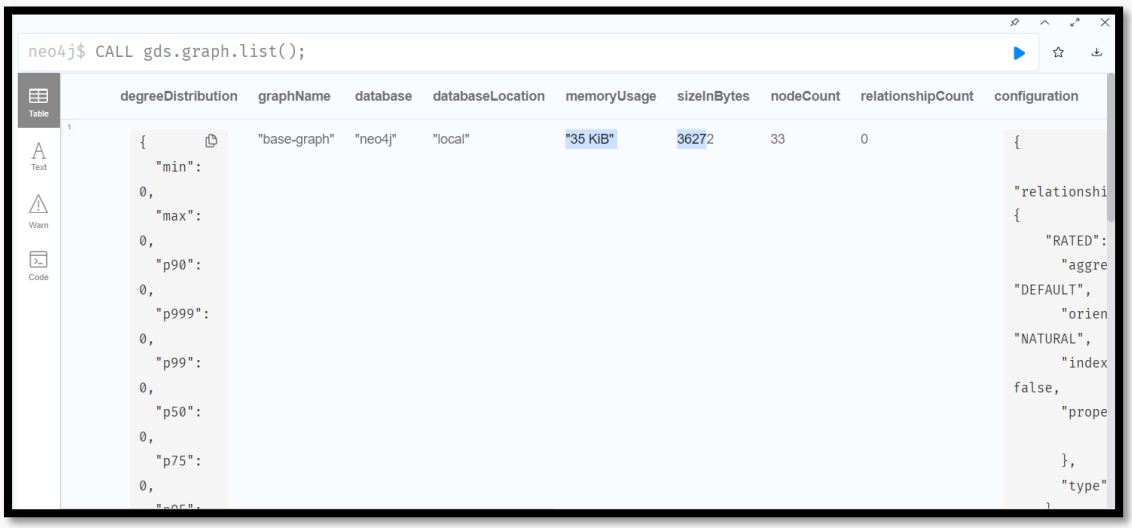
Appliquer les concepts de projections natives et explorer les impacts de l'orientation des relations sur les analyses graphiques.**Consignes :**

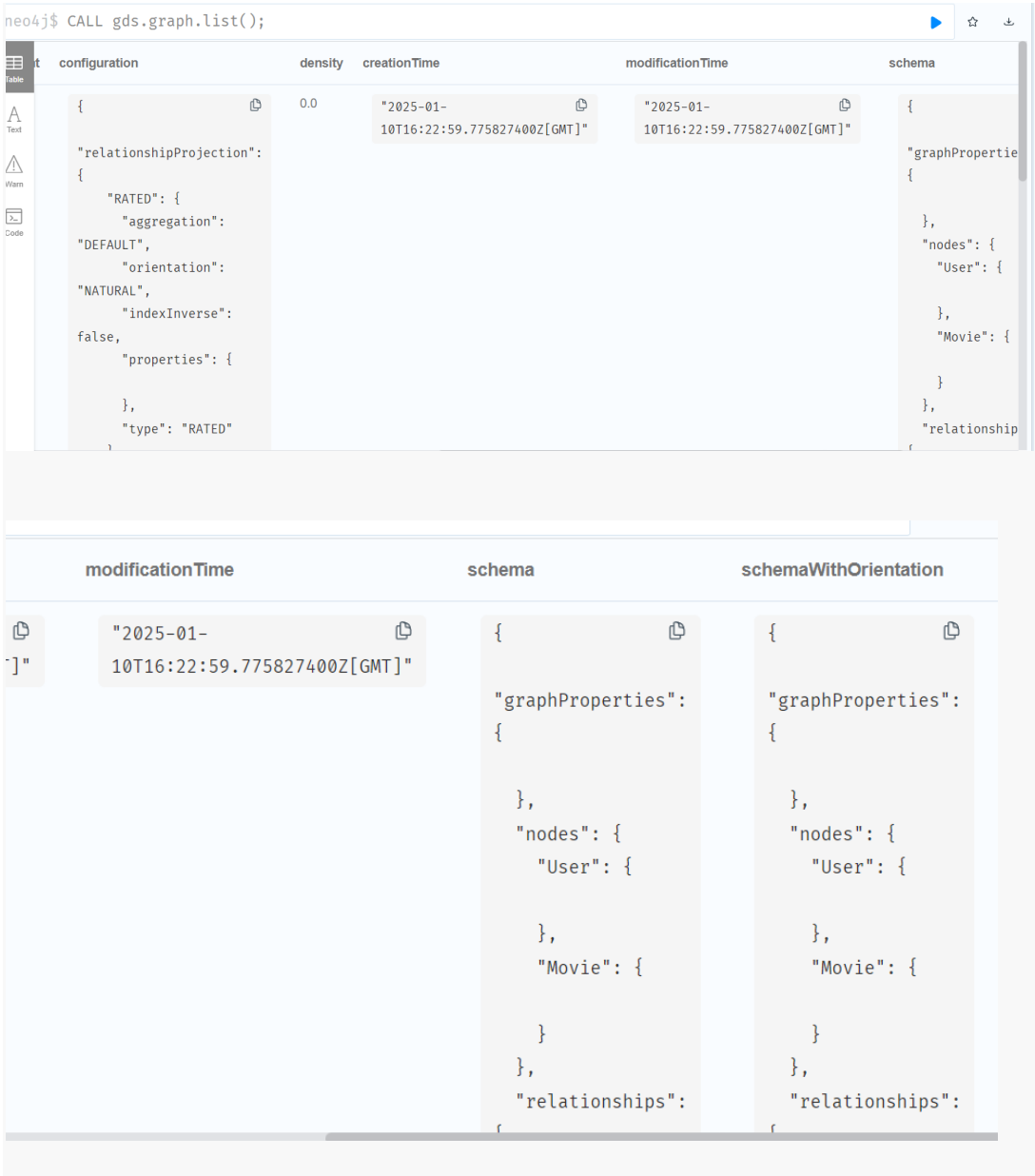
Étape 1 : Création d'une projection de base

1. Projetez un graphe contenant les nœuds `User` et `Movie` ainsi que les relations `RATED`.



2. Listez les graphes projetés pour vérifier la projection.





3. Utilisez l'algorithme `degree` pour calculer le nombre de connexions de chaque nœud, et affichez les résultats pour les nœuds `Movie`.

```
CALL gds.degree.stream('base-graph')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).title AS movieTitle, score AS degree
ORDER BY degree DESCENDING
LIMIT 10
```

	movieTitle	degree
1	"Inception"	0.0
2	"The Dark Knight"	0.0
3	"The Matrix"	0.0
4	"Inception"	0.0
5	"The Dark Knight"	0.0
6	"The Matrix"	0.0
7		

5	"The Dark Knight"	0.0
6	"The Matrix"	0.0
7	"Inception"	0.0
8	"The Dark Knight"	0.0
9	"The Matrix"	0.0
10	"The Matrix"	0.0

Observation :

- Les résultats montreront un degré de 0 pour les films, car la direction des relations dans la base de données (`User → Movie`) empêche le calcul correct.

Étape 2 : Modification de l'orientation des relations

1. Projetez un graphe où la relation `RATED` est inversée (`RATED_BY`).

```
CALL gds.graph.drop('base-graph', false);

CALL gds.graph.project(
  'reverse-graph',
  ['User', 'Movie'],
  {RATED_BY: {type: 'RATED', orientation: 'REVERSE'}}
);
```

```
neo4j$ CALL gds.graph.drop('base-graph', false)
neo4j$ CALL gds.graph.project( 'reverse-graph', ['User', 'Movie'], {RATED_BY: {type: 'RATED', or…
```

2. Réutilisez l'algorithme `degree` pour calculer combien de fois chaque film a été noté.

```
CALL gds.degree.stream('reverse-graph')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).title AS movieTitle, score AS ratingCount
ORDER BY ratingCount DESCENDING
LIMIT 10;
```

	movieTitle	ratingCount
1	"Inception"	0.0
2	"The Dark Knight"	0.0
3	"The Matrix"	0.0
4	"Inception"	0.0
5	"The Dark Knight"	0.0
6	"The Matrix"	0.0
7		

3. Comparez les résultats avec ceux obtenus lors de la projection de base.

- Les résultats sont le même dans les deux projections.

Étape 3 : Relations non orientées

1. Projetez un graphe en spécifiant que les relations `RATED` sont non orientées.

```
CALL gds.graph.project(
    'undirectedGraph',
    ['User', 'Movie'],
    {
        RATED: {
            orientation: 'UNDIRECTED'
        }
    }
);
```

nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
{ "User": { "label": "User", "properties": { } }, "Movie": { "label": "Movie", "properties": { } } }	{ "RATED": { "aggregation": "DEFAULT", "orientation": "UNDIRECTED", "indexInverse": false, "properties": { }, "type": "RATED" } }	"undirectedGraph"	33	0	187

arted streaming 1 records after 21 ms and completed after 378 ms.

2. Appliquez l'algorithme `degree` pour analyser les connexions dans ce graphe.

```
CALL gds.degree.stream('undirectedGraph')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).title AS Node, score
ORDER BY score DESCENDING;
```

Table | Text | Code

	Node	score
1	"The Matrix"	0.0
2	"Inception"	0.0
3	"The Dark Knight"	0.0
4	"The Matrix"	0.0
5	"Inception"	0.0
6	"The Dark Knight"	0.0
7		

Started streaming 33 records after 35 ms and completed after 172 ms.

Étape 4 : Analyse avancée

1. Ajoutez une propriété `weight` aux relations `RATED`.

```
MATCH (u:User)-[r:LIKES]->(m:Movie)
SET r.weight = toInteger(rand() * 10);
```

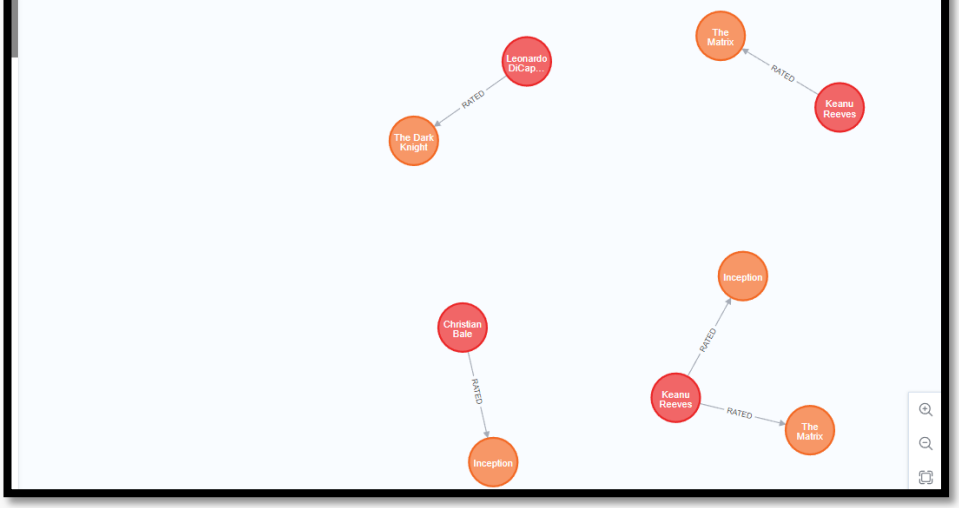
2. Projetez un graphe en incluant cette propriété et utilisez-la dans un calcul de degré pondéré

```
CALL gds.graph.drop('undirected-graph', false);

CALL gds.graph.project(
  'weighted-graph',
  ['User', 'Movie', 'Person'],
  {LIKES: {type: 'LIKES', orientation: 'NATURAL', properties: '
    weight'}}
);
```

```
CALL gds.degree.stream('weighted-graph', { relationshipWeightProperty: 'weight'})
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).title AS nodeName, score AS weightedDegree
ORDER BY weightedDegree DESCENDING
LIMIT 10;
```

```
MATCH (u:User)-[r:RATED]→(m:Movie)
RETURN u, r, m
LIMIT 10;
```



3. Comparez les résultats avec ceux des étapes précédentes.

- **Le degré simple est une mesure brute du nombre d'interactions, tandis que le degré pondéré fournit une vue plus nuancée de la qualité de ces interactions.**

- **Questions analytiques :**

  — **Quel type d'orientation (naturelle, inversée ou non orientée) fournit les informations les plus pertinentes pour analyser les films les plus notés?**

  L'orientation inversée est la plus pertinente pour analyser les films les plus notés, car elle permet de compter combien de fois un film a ´été notés par des utilisateurs.

  — **Quels sont les avantages d'utiliser des relations non orientées ou inversées dans des analyses spécifiques?**

Les relations non orientées sont utiles pour des analyses ou` la direction des relations n'est pas importante (par exemple, les réseaux sociaux). Les relations inversées sont utiles pour analyser les relations dans le sens opposé (par exemple, les films notés par les utilisateurs).

— **Proposez une situation ou` l'ajout d'une propriété pondérée pourrait améliorer les résultats d'analyse.**

- Dans un système de recommandation, l'ajout d'une propriété pondérée (comme la note donnée par un utilisateur) permet de donner plus de poids aux films bien notés, améliorant ainsi la qualité des recommandations