

TP Neo4j

MUSTAPHA OULGHAZI

Étape 1 : Création d'une projection de base

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

```
1 CALL gds.graph.project(
2   'baseGraph',
3   ['User', 'Movie'],
4   ['RATED']
5 );
```

	nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1	<pre>{ "User": { "label": "User", "properties": { } }, "Movie": { "label": "Movie", "properties": { } } }</pre>	<pre>{ "RATED": { "aggregation": "DEFAULT", "orientation": "NATURAL", "indexInverse": false, "properties": { }, "type": "RATED" } }</pre>	"baseGraph"	40	770	339

2 :Listez les graphes projetés pour vérifier la
projection.

```
neo4j$ CALL gds.graph.list();
```

	degreeDistribution	graphName	database	databaseLocation	memoryUsage	sizeInBytes	nodeCount	relationshipCount	configuration	density	creationTime
Text	{ "min": 0, "max": 55, "p90": 52, "p999": 55, "p99": 55, "p50": 0, "p75": 40, }	"baseGraph"	"neo4j"	"local"	"303 KiB"	310408	40	770	{ "relationshipProjection": { "RATED": { "aggregation": "DEFAULT", "orientation": "NATURAL", "indexInverse": false, "properties": { }, "type": "RATED" } } }	0.4935897435897436	"2025-01-14T21:53:55"

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

```
1 CALL gds.degree.stream('baseGraph')
2 YIELD nodeId, score
3 WHERE gds.util.asNode(nodeId):Movie
4 RETURN gds.util.asNode(nodeId).title AS movie, score
5 ORDER BY score DESC;
```

	movie	score
1	"Inception"	0.0
2	"The Matrix"	0.0
3	"Inception"	0.0
4	"The Matrix"	0.0
5	"Inception"	0.0
6	"The Matrix"	0.0

Étape 2 : Modification de l'orientation des relations

1 :

```
1 \CALL gds.graph.project(
2   'reversedGraph',
3   ['User', 'Movie'],
4   { RATED: { orientation: 'REVERSE' } }
5 );
```

	nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1	{ "User": { "label": "User", "properties": {} }, "Movie": { "label": "Movie", "properties": {} } }	{ "RATED": { "aggregation": "DEFAULT", "orientation": "REVERSE", "indexInverse": false, "properties": {} }, "type": "RATED" }	"reversedGraph"	40	770	14

2-3 :

```
1 CALL gds.degree.stream('reversedGraph')
2 YIELD nodeId, score
3 WITH gds.util.asNode(nodeId) AS node, score
4 WHERE labels(node) = ['Movie']
5 RETURN node.title AS movie, score
6 ORDER BY score DESC;
```

	movie	score
1	"Inception"	55.0
2	"The Matrix"	55.0
3	"Inception"	54.0
4	"The Matrix"	54.0
5	"Inception"	52.0
6	"The Matrix"	52.0

Étape 3 : Relations non orientées

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

```
1 CALL gds.graph.project(
2   'undirectedGraph',
3   ['User', 'Movie'],
4   { RATED: { orientation: 'UNDIRECTED' } }
5 );
```

	nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1	{ "User": { "label": "User", "properties": { } }, "Movie": { "label": "Movie", "properties": { } } }	{ "RATED": { "aggregation": "DEFAULT", "orientation": "UNDIRECTED", "indexInverse": false, "properties": { }, "type": "RATED" } }	"undirectedGraph"	40	1540	13

- 3 : Appliquez l'algorithme degree pour analyser les connexions dans ce graphe.

```
1 CALL gds.degree.stream('undirectedGraph')
2 YIELD nodeId, score
3 RETURN gds.util.asNode(nodeId).title AS entity, score
4 ORDER BY score DESC;
```

	entity	score
1	<i>null</i>	55.0
2	<i>null</i>	55.0
3	"Inception"	55.0
4	"The Matrix"	55.0
5	<i>null</i>	54.0
6	<i>null</i>	54.0

Étape 4 : Analyse avancée

1. Ajoutez une propriété weight aux relations RATED.

```
1 MATCH ()-[r:RATED]→()
2 SET r.weight = r.rating;
```

Table

Code

Set 770 properties, completed after 243 ms.

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

```
1 CALL gds.graph.project(
2   'weightedGraph',
3   ['User', 'Movie'],
4   {
5     RATED: {
6       orientation: 'NATURAL',
7       properties: 'weight'
8     }
9   }
10 )
```

	nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1	{ "User": { "label": "User", "properties": { } }, "Movie": { "label": "Movie", "properties": { } } } }	{ "RATED": { "aggregation": "DEFAULT", "orientation": "NATURAL", "indexInverse": false, "properties": { "weight": { "aggregation": "DEFAULT", "property": "weight", "defaultValue": null } } }, "type": "RATED" }	"weightedGraph"	40	770	61

3 :Comparez les résultats avec ceux des étapes précédentes.

```

1 CALL gds.degree.stream('weightedGraph', { relationshipWeightProperty: 'weight' })
2 YIELD nodeId, score
3 WITH gds.util.asNode(nodeId) AS node, score
4 WHERE 'Movie' IN labels(node)
5 RETURN node.title AS movie, score
6 ORDER BY score DESC;

```

	movie	score
1	"Inception"	0.0
2	"The Matrix"	0.0
3	"Inception"	0.0
4	"The Matrix"	0.0
5	"Inception"	0.0
6	"The Matrix"	0.0