# Étape 1 : Projection de base

Nous avons commencé par projeter un graphe contenant les nœuds User et Movie, ainsi que les relations RATED :

cypher

CopierModifier

```
CALL gds.graph.project(
  'base-graph',
  ['User', 'Movie'],
  ['RATED']
);
```

## Liste des graphes projetés

Pour vérifier les graphes projetés, la commande suivante a été utilisée :

cypher

CopierModifier

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

## Calcul du degré avec l'algorithme degree

Nous avons appliqué l'algorithme degree pour calculer le nombre de connexions de chaque nœud dans le graphe projeté :

cypher

CopierModifier

```
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;

Observation

Les résultats ont montré un degré de **0 pour les films**. Cela est dû à la direction des relations dans la base de données (User → Movie), qui empêche un calcul correct du degré des films.

# Étape 2 : Modification de l'orientation des relations

Pour résoudre ce problème, nous avons inversé l'orientation des relations. Un nouveau graphe a été projeté où les relations RATED ont été orientées en REVERSE :

cypher

CopierModifier

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


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

## Calcul du degré avec degree sur le graphe inversé

Après inversion, l'algorithme degree a permis de calculer combien de fois chaque film a été noté :

cypher

CopierModifier

```
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;
```

# Étape 3 : Relations non orientées

Pour une analyse plus globale, nous avons projeté un graphe où les relations RATED sont **non orientées** :

cypher

CopierModifier

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


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

## Calcul du degré pour les relations non orientées

L'algorithme degree a été utilisé à nouveau :

cypher

CopierModifier

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

# Étape 4 : Analyse avancée avec une propriété pondérée

Pour aller plus loin, nous avons ajouté une propriété weight aux relations RATED et projeté un graphe pondéré :

cypher

CopierModifier

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


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

### Calcul du degré pondéré

L'algorithme degree a été appliqué en tenant compte de la propriété weight :

cypher

CopierModifier

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

## Questions analytiques

1. **Quel type d'orientation fournit les informations les plus pertinentes pour analyser les films les plus notés ?**
   - **Orientation naturelle** (User → Movie) : Pertinente pour analyser la popularité en fonction des notes des utilisateurs.
   - **Orientation inversée** (Movie → User) : Utile pour déterminer combien de fois un film a été noté.
   - **Relations non orientées** : Plus générales, adaptées à une vue d'ensemble.
2. **Avantages d'utiliser des relations non orientées ou inversées :**
   - **Non orientées** : Idéales pour des analyses où la direction des relations n'a pas d'importance.
   - **Inversées** : Permettent d'étudier les films du point de vue des utilisateurs (ex. fréquence de notation).
3. **Ajout d'une propriété pondérée :**
   Une propriété pondérée peut améliorer l'analyse en attribuant plus de poids à des relations

importantes, comme des notes élevées. Cela permet de mettre en avant les films les plus appréciés au lieu de se concentrer uniquement sur le volume de notations.

## Conclusion

Ce TP a permis de comprendre les effets des différentes orientations des relations sur les analyses dans Neo4j GDS. L'ajout de propriétés pondérées a montré comment enrichir l'analyse en tenant compte de l'intensité des relations.