Graph Projection and Analysis in Neo4j

Introduction

This document provides an example of creating a graph projection in Neo4j using the Graph Data Science (GDS) library. We project a subgraph containing Person and Loan nodes, connected by the APPLIES_FOR relationship. The projected graph is then analyzed using PageRank.

Step 1: Define the Graph Model

Nodes:

- Person: Represents applicants for loans.

- Loan: Represents loans being applied for.

Relationships:

- APPLIES_FOR: Connects Person to Loan.

Step 2: Import the Data into Neo4j

Ensure the data is imported into Neo4j using LOAD CSV commands. Create Person and Loan nodes, and APPLIES_FOR relationships connecting them.

Step 3: Create the Graph Projection

Use the following Cypher command to create a graph projection in-memory:

```
CALL gds.graph.project(
    'loanGraph',
    ['Person', 'Loan'],
    {
        APPLIES_FOR: {
            orientation: 'NATURAL'
```

```
    }
  }
);
```

Step 4: Verify the Projection

Use the following command to verify the projection:

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

Step 5: Run a Graph Algorithm

Run the PageRank algorithm to rank nodes based on importance:

```
CALL gds.pageRank.stream('loanGraph')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name AS name, score
ORDER BY score DESC
LIMIT 10;
```

Step 6: Drop the Projection

After completing the analysis, drop the in-memory graph projection to free resources:

```
CALL gds.graph.drop('loanGraph');
```

Conclusion

This example demonstrates how to create, verify, and analyze a graph projection in Neo4j using the GDS library. Graph projections allow for efficient computations on a subset of the data.