Exercise #8

Submission:

Submit your exercise as a SINGLE ZIP file on Canvas by the due date.
Your submitted ZIP file must have the name:
Exercise_8_Your_LastName.zip

Deliverables:

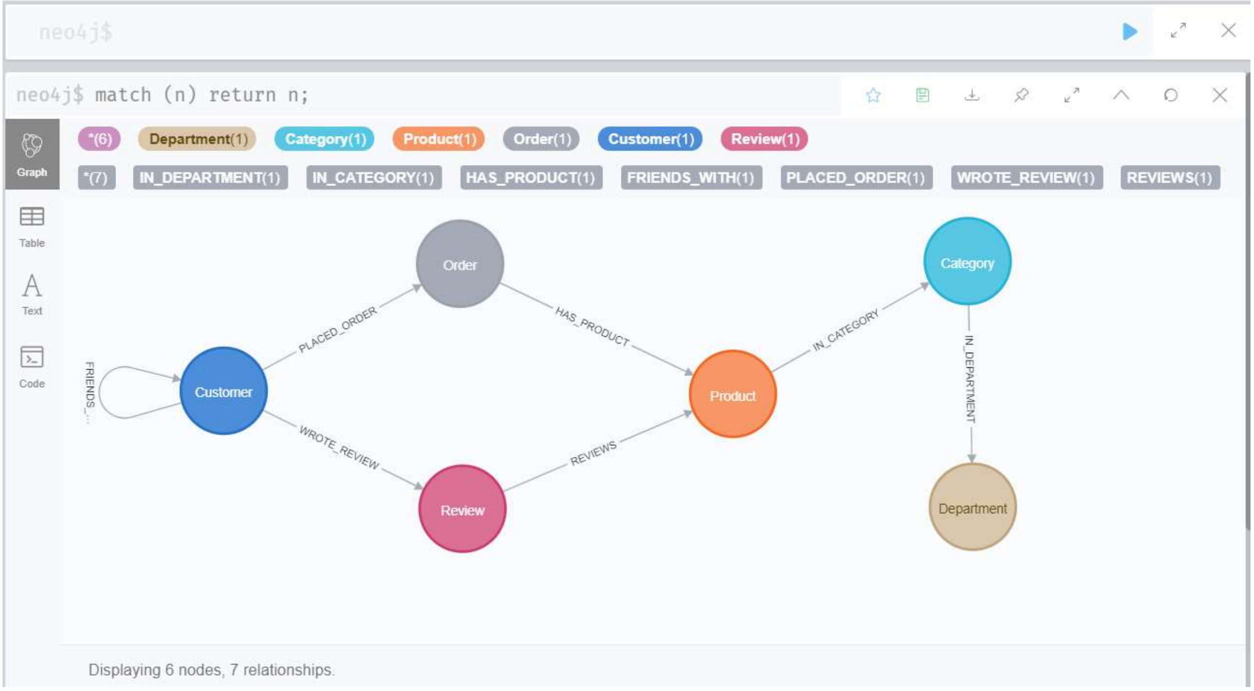
Your ZIP file for the exercise submission must include the following:

- All source code that you installed, compiled and built on your personal computer.
- Panopto video recording of a live run of your code on your personal development computer.

Requirements Specification:

Download, and install Neo4j 4.2.1 on your personal development computer.

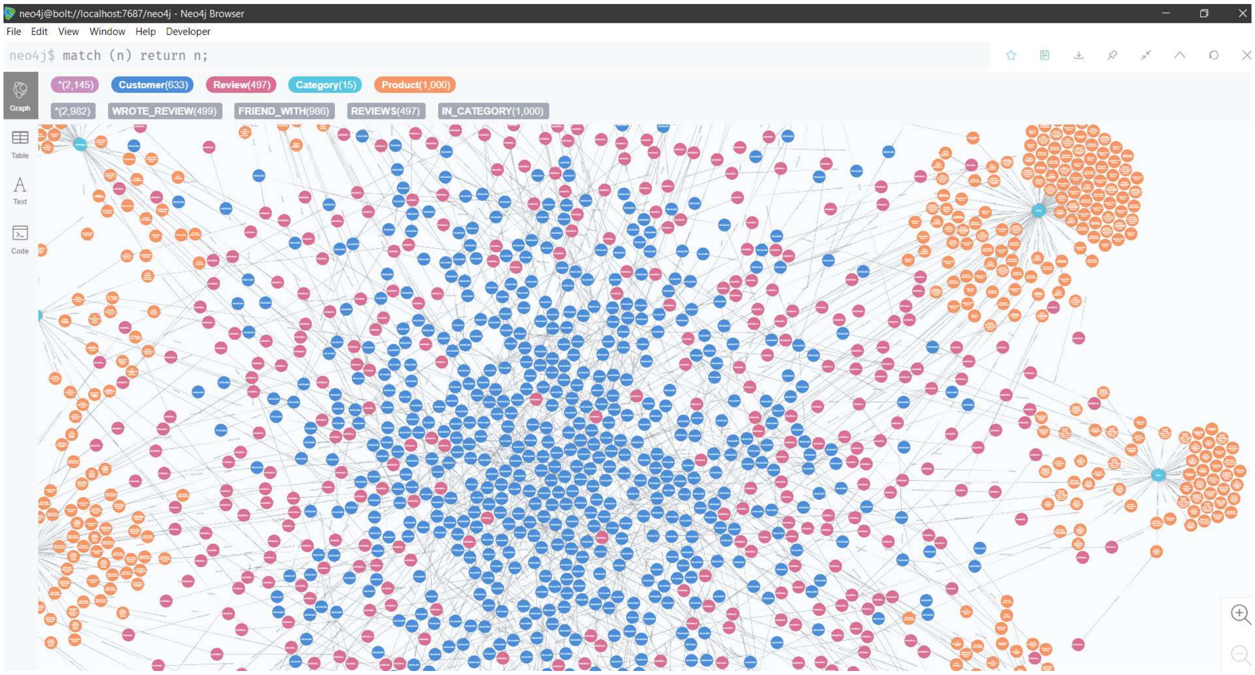
After you install Neo4j on your personal development computer, you will create the following Customer-Friends-Product-Reviews Graph Database for OnMart Super Store and use the **Betweenness** algorithm in order to find the **Superconnectors** in its graph database (shown in next page) based on the following Graph Data Model :



```
neo4j$ match (n) return n;
```

Displaying 6 nodes, 7 relationships.

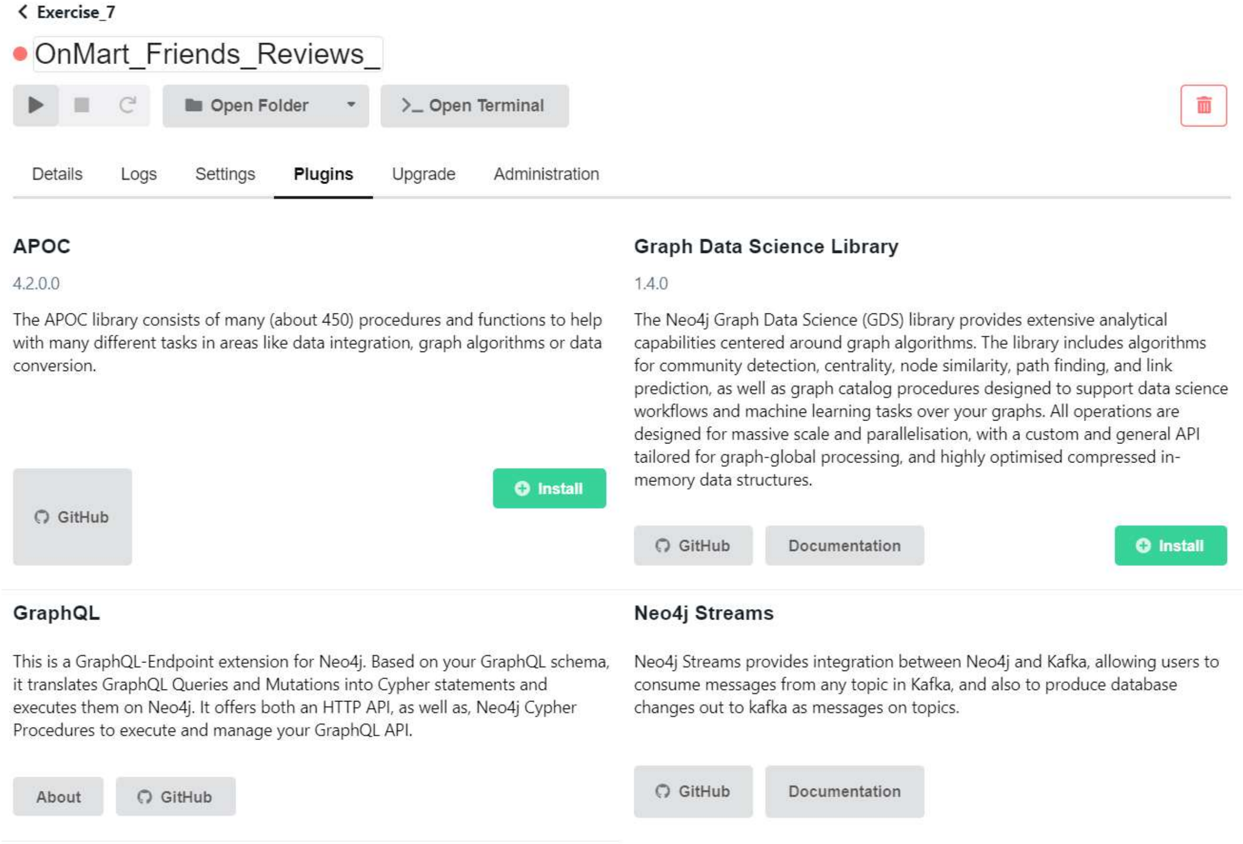
Graph Database



```
neo4j$ match (n) return n;
```

Create the Customer-Friends-Product-Reviews Graph Database for OnMart

1. Startup Neo4j
2. Use Cypher to create the Customer-Friends-Product-Reviews graph database for OnMart
3. All Cypher code must be saved in a file with the name cypher.txt
4. After you create OnMart_Friends_Reviews_PageRank graph database, go to the plugin tab (see below screenshot) and install the following plugins
    1) APOC
    2) GDS - Graph Data Science Library



APOC

4.2.0.0

The APOC library consists of many (about 450) procedures and functions to help with many different tasks in areas like data integration, graph algorithms or data conversion.

Graph Data Science Library

1.4.0

The Neo4j Graph Data Science (GDS) library provides extensive analytical capabilities centered around graph algorithms. The library includes algorithms for community detection, centrality, node similarity, path finding, and link prediction, as well as graph catalog procedures designed to support data science workflows and machine learning tasks over your graphs. All operations are designed for massive scale and parallelisation, with a custom and general API tailored for graph-global processing, and highly optimised compressed in-memory data structures.

GraphQL

This is a GraphQL-Endpoint extension for Neo4j. Based on your GraphQL schema, it translates GraphQL Queries and Mutations into Cypher statements and executes them on Neo4j. It offers both an HTTP API, as well as, Neo4j Cypher Procedures to execute and manage your GraphQL API.

Neo4j Streams

Neo4j Streams provides integration between Neo4j and Kafka, allowing users to consume messages from any topic in Kafka, and also to produce database changes out to kafka as messages on topics.

5. Write and execute Cypher statements to create the customer label/nodes
6. Write and execute Cypher statements to create the FRIEND_WITH relationships/type
7. Write and execute Cypher statements to create the Review label/nodes
8. Write and execute Cypher statements to create the WROTE_REVIEW relationships/type
9. Write and execute Cypher statements to create the Product label/nodes
10. Write and execute Cypher statements to create the IN_CATEGORY relationships/type
11. Write and execute Cypher statements to create the REVIEWS relationships/type
12. Write and execute Cypher statements using **Betweenness** to find top 10 **Superconnectors**