Big Graph Processing Systems 2025

TP Component 1: Schema Discovery

Christopher Spinrath

General Information

The goal of this component is to obtain a schema for the given property graph. It consists of six tasks. The first task is to install Neo4j and import the database.
A complete solution for tasks 2 to 5 consists of

- an openCypher query,
- the number of answers it returns, and
- an explanation of the query.

For the query you can gain up to 2 points, for the number of answers 1 point, and for the explanation 1 point. The last step is to illustrate the property graph schema, for which you can gain up to 8 points. The total amount of points achievable with this first component is thus 24 points.

Important

- Do not use `CALL ... YIELD` clauses to call procedures. They are Neo4j specific and are not part of openCypher/GQL.
- Make sure that all queries are free of syntax errors by running them, even if you make only small changes. A query with syntax errors will be graded with 0 points.
- A good explanation helps someone familiar with basic knowledge on openCypher to understand the (key) idea(s) of a query. "Reading" or repeating the query in natural language is, for example, not very helpful.
- Solve the tasks yourself, on your own (that is, there are no group submissions). **Plagiarism is fraud.**

Task 1 **(0 points)**

a) Install Neo4j.

- Make sure you are using **version 5.26 or newer**.
- Make sure you have enough memory, at least 20GB.
- A simple option is to install Neo4j Desktop by following the official instructions: `https://neo4j.com/docs/desktop-manual/current/installation/`.

b) Download and import the database.

- Download the database dump file from `https://partage.liris.cnrs.fr/index.php/s/T2RzHWYEKwTcjLG`.
- Import the dump file by following the official instructions: `https://neo4j.com/docs/desktop-manual/current/operations/create-from-dump/`.
- The graph consists of 2 016 523 nodes and 3 339 267 relationships.

Task 2 **(4 points)**

a) Write an openCypher query that returns all distinct node labels, alphabetically ordered.
Note: The first row should consist of the string `"Address"`, not of the list `["Address"]`.

b) How many answers does your query return?

c) Explain concisely how you came up with your query.

Task 3 **(4 points)**

a) To obtain information about the hierarchy of labels, write an openCypher query that returns

- each distinct label L with
- the list of labels that can occur together with L at the same node, and
- the size s of this list

ordered by the label L and the size s.
For example, the row

```
"Address" | ["Address"] | 1
```

should be one of the answers returned by the query.
Hint: Extend your previous query with another `MATCH` (and `RETURN`) clause.

b) How many answers does your query return?

c) Explain concisely how you came up with your query.

Task 4 **(4 points)**

a) Write an openCypher query that returns

- each distinct label L with
- the list of relationship types of outgoing edges from nodes labelled with L, and
- the size of this list.

For example, the row

```
"Address" | ["same_as", "same_address_as"] | 2
```

should be one of the answers returned by your query.
Hint: You can again extend your first query by another MATCH (and RETURN) clause.

b) How many answers does your query return?

c) Explain concisely how you came up with your query.

Task 5 **(4 points)**

a) Write an openCypher query that returns each distinct combination of

- a property key/name L with
- the list of labels of nodes that have property K

ordered by the size of the label list.
For example, the row

```
"struck_off_date" | ["Entity", "StruckOff", "Other"]
```

should be one of the answers returned by your query, and means that nodes with the label Entity, StruckOff, or Other can have a property `struck_off_date`.
Hint: A possible approach is to unwind lists of labels and properties and then use an aggregation.

b) How many answers does your query return?

c) Explain concisely how you came up with your query.

Task 6 **(8 points)**

Illustrate a property graph schema taking into account

- node labels and relationship types,
- properties and their types,
- label hierarchies (for instance, you can use edges labelled `subtype_of`).

Properties, that every node can have, can be described separately. It is not required to indicate whether a property is optional or not.

You can use a drawing tool on your computer, or draw it by hand on paper and take a (readable) photo.

Of course, you can write further openCypher queries to obtain more knowledge about the schema.