Neo4j processing

**Data sources**

The contribution data came from a powerBI database. Access can be shared with anyone with a powerBI account. Harriet Kenny gave me access.

Different databases and filters are available, I pulled the ±30 000 rows which were readily available as a csv.

Research methods data came from Jim. There are two available sets, one for education and one for psychology. I’m not sure about their origins. Used Jim’s preprocessing script to extract DOIs and research methods from ±300 000 rdf files from the psychology folder. Then built a dictionary, with research method keys mapped to an array of DOIs (which are tagged by that method)

All the data came from these two sources.

**Preprocessing for neo4j**

Filtered the contribution data using the DOIs contained in the research method data. The result was ±400 rows of contribution data, which would correspond to at least one research method.

Iterated through the {method : [DOIs]} dictionary to build a dataframe of unique rows of a research method and the contribution data of each article tagged by that method. This is ±32 000 rows, with 518 unique research methods and ±400 unique DOIs. This contains all the datapoints needed for importing graph data into neo4j.

Preprocessing code: <https://github.com/sagepublishing/research-impact/blob/master/SRM_csv.ipynb>



**Neo4J**

Is a graph database which allows import and modelling of data. The key features of graph data are nodes and relationships. A single node has a type and a token, where the token can be described by a set of attributes. Relationships describe how a node is connected to others. Once data is imported, CYPHER queries allow nodes and relationships to be inspected.

Neo4j can run from the desktop, where projects are initiated and saved and launched in the neo4j browser.

**Importing data**

There appear to be two main approaches to importing data into neo4j. A complete csv can be LOADED and then queried using CYPHER, or LOAD CSVs containing the precise nodes and relationships you’re interested in.

I chose the latter because I’ve spent way more time using Python and Pandas than Neo4j and CYPHER. This meant creating 5 CSVs in total:

LOAD methods.csv: generate *method* nodes, with a method label (*research report*, *sampling*, etc.,) as an attribute.

LOAD articles.csv: generate *article* nodes, with title, DOI and journal as attributes.

LOAD authors.csv: generate *author* nodes, with their institute as an attribute

LOAD article -> method.csv: generate relationship -[:Applies]-> mapping articles to research methods

LOAD author -> article.csv: generate relationship -[:Contributes]-> mapping authors to articles.

Pretty much just followed the tutorial: <https://www.youtube.com/watch?v=1U6iUTV_Dco&t=348s>