Graph Theory Project 2017

*Kelvin Murphy*

*G00311695*

Design and prototype a Neo4j database for use in a timetabling system for GMIT.

For my third year project in the module Graph Theory we have been asked to detail and create a working prototype of a timetabling system for a third level institute, using a prototype neo4j database. I have chosen to detail a plan and create my graph-based database on the GMIT Galway Campus and more specifically Third year software development, which is my own class.

I thought that by using my own timetable it would feel more familiar to me and I would find it more comfortable working with it as I could see nodes as more than just information bubbles and see them as people and classrooms. By seeing them as real objects it will make it easier for me to rationalize them and store them in the database.

The reason why I am going to recreate the timetable using a graph based database is to show what I have learned over the semester in this module and by detailing and creating a graph database I am showing that I understand the concepts of Graph Theory, and that I am able to solve problems using graph based solutions, both simple and complex in this module by what I have learned over the last semester.

I will update and commit my written document outlining everything I plan to do and my actual project to github on a regular basis as to keep all of my work up to date and also it is a handy back up because if my computer crashes I will have all my information on github and can simply take it down off of github on to another computer and continue my work.

I have decided to use Microsoft word to write my document and Neo4j community edition to create the graph database. I have never used neo4j before so this will be and interesting earning experience for me.

My database will store all the necessary data to form a working usable timetable. It will store the rooms as nodes so they can be easily attached to on the graph to other nodes. It will store the three different classes as nodes so they can be connected to other nodes. It will store the time slots and the sate slot, again as nodes so that they can be joined up and used with other nodes. The database will store the lecturers as nodes to make sure they are not otherwise busy and can be put in with a certain class at a certain time. Lastly it will store the subjects (modules) for the course, again these will also be stored as nodes so that all of the above, the rooms, class groups, time/date slots, lecturers and subjects can all be interconnected on the graph and therefore create a graph that will allow all of these to work together without overbooking any one node.

All of these nodes then interconnect to create a complex graph that I will use neo4j community edition to create. It will then store as many nodes as I add to it but for this timetable project the only nodes I need to use in order to fill out the graph and timetable are the ones I listed below.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Rooms | Node | | Class Groups | Node | | Time/Date Slot | Node | | Lecturers | Node | | Subjects | Node | |

A simple diagram shows how these are stored, keep in mind that these are only the main headers and each of the above has many nodes not just one.

A more detailed sub diagram to show all the sub categories under the main ones will be more helpful into understanding how these are stored so I will create a table of all the sub categories as there are a numerous amounts of each of the above nodes.

|  |  |
| --- | --- |
| **Group** |  |
| A | Node |
| B | Node |
| C | Node |
| **Lecturer** |  |
| Ian Mc’Loughlin | Node |
| Martin Hynes | Node |
| Damien Costello | Node |
| Patrick Moran | Node |
| Gerard Harrison | Node |
| Deidre O’Donovan | Node |
| **Subject** |  |
| Graph Theory | Node |
| Database Management | Node |
| Server Side Rad | Node |
| Mobile Applications | Node |
| Software Testing | Node |
| **Rooms** |  |
| 145 | Node |
| 994 | Node |
| 223 | Node |
| PF 05 | Node |
| 481 CR4 | Node |
| 436 CR5 | Node |
| 482 CR3 | Node |
| 470 | Node |
| 379 | Node |
| 162 | Node |
| 938 | Node |
| 208 | Node |
| 997 | Node |
| 939 | Node |
| 995 | Node |
| PF 18 | Node |
| 483 CR2 | Node |
| **Time/Date** |  |
| Monday – Friday  9:00 – 18:00 | Every Hour here is a node |
|  |  |

The Database will store all of this information.

Its interesting how Neo4j works by creating a graph based database and could come in very useful in lots of different types of databases but for me and right now the timetable seems to be best suited to use the Neo4j community edition to create a graph for it. It is designed around relationships with its nodes and it is the most popular of all the NoSql databases. It has huge scalability that can reach billions of nodes, however I won’t be using near that amount, a site however that would use that many would be Facebook.

Neo4j works by retrieving information based on interconnections between data rather than the orthodox Sql where it runs through the data systematically to find the information. It is and open source and is supported by Neo Technology hence the name Neo4j (Neo for java). Directional relationships in graph databases have properties that describe them and all nodes are connected to each other making the directional relationships work.

Simply put Neo4J finds relationships (connections) between data and in turn finds their value.