COMP5338: Assignment 2

Neo4j – Modelling Courses, Units and Students in a Graph Database

Jesse Serina Narvasa – jnar3156

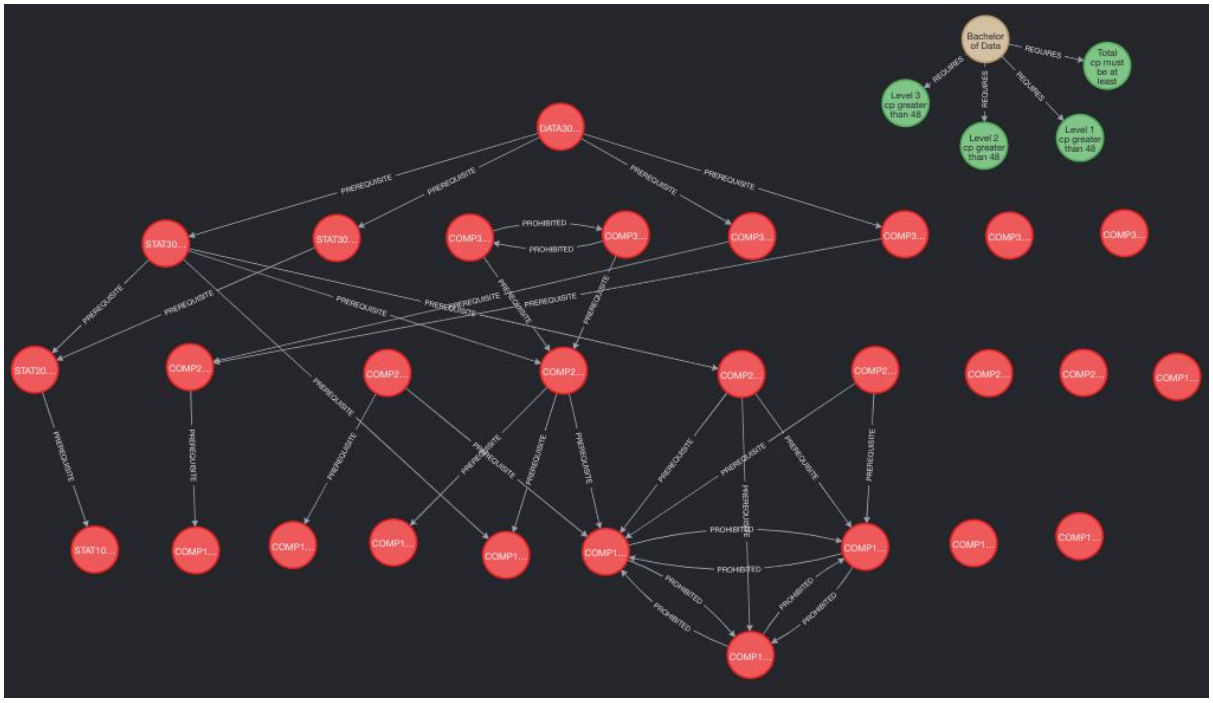

Figure 1 Screenshot of the graph illustrating the Course, its Requirements, and the Units available within the Course

The screenshot above demonstrates the relationships between the unit nodes, stored with the label **Unit** and illustrated above with the colour red. Each unit node contains the properties: code, cp, level, and title.

Unit nodes can also have relationships with other unit nodes, as seen above. The first type is a **PREREQUISITE** relationship, where the unit pointing in the direction of another unit means that the unit being pointed to is a prerequisite that needs to be taken beforehand.

Some units however have prerequisites which are linked with *OR* operators, and this is supported by the graph through the use of the property "alternate" within the PREREQUISITE relationship. The alternate property supports a list and specifies the code/s of the other unit which can be used as an alternative to the current unit being pointed at. This means that unit A with a PREREQUISITE relationship to unit B, with the relationship having the property called "alternate" containing a list of unit codes C and D; it means that in order to take unit A, the student must have taken either unit B, C, or D.

Likewise, there are units which have PREREQUISITE relationships which is a combination of *AND and OR*. There is no explicit property that denotes AND in the way the graph is structured, unlike OR which has the alternate property to store the list of unit codes. Instead, when unit A has a PREREQUISITE relationship to unit E, and this relationship doesn't have either units B, C, or D listed in its "alternate" property, then it means that unit E must

be taken in order to be eligible to take unit A, along with meeting the other set of prerequisites previously discussed which is linked with OR operators.

Finally, there is the **PROHIBITED** relationship which is the second type of relationship between Unit nodes. This relationship describes that the student cannot enrol in the unit, if he/she has completed a unit which has a PROHIBITED relationship with the unit they're enrolling into.

The graph above also includes the **Degree** node, which represents the degrees possible to be studied. In this graph, we only have one degree available, which is the "Bachelor of Data". The degree node contains the properties of code and its name only. The degree node also has requirements for the student in order for the degree to be completed, and this is modelled as a relationship of type **REQUIRES** to the Requirement nodes.

The **Requirement** nodes simply represent the different requirements that a degree might have that a student would need to complete. It includes information, stored as a property such as its name, cp, and level. The name is a descriptive name on what the requirement is about, cp states the amount of credit points is required to be met, and level is the level of study that the amount in cp relates to e.g., student must complete 48 credit points in level 2000 units

The screenshot of the graph above doesn't include the students, whom are represented with the label **Person**. Firstly, the reason students are classified with the label Person is so that the graph is versatile in the future. In the event that there is a teacher, then the teacher can also be represented with the label Person. Moreover, if the teacher is also a student, then that person doesn't have to be represented twice since the label Person would be versatile enough for that, and hence Person is the name of the label. The person label only contains the properties name, and also current_degree (since the only persons in this graph are students, no other properties have been added). To keep things simple, it is assumed that you can only study ONE degree at a time, and hence, current degree property has the expected value of a string denoting the degree's code. Moreover, queries within the browser will expect current_degree to only contain string as previously mentioned. The degree that the person is studying Is marked with the relationship type **STUDIES** and does not have any properties.

A student, represented with the label Person, can be enrolled in many units, and is shown using the relationship type **ENROLLED_IN**. This relationship has the property "status" and has been included to future-proof the graph in the event that there are instances of students added later whom enrols in a unit but does not complete it. However, since that is not a requirement within the specifications of this assignment, the queries within the browser makes the assumption that a person with an ENROLLED_IN relationship to a unit means that they have passed it. Hence, the property "status" at present, is redundant due to this assumption with only two students having passed all units they've enrolled in.