Prashant B. Bhuyan
is607 Week12Quiz Neo4J

Problem 1)

Pre-Processing:

I separated out students by Enrolled (Grade = ‘IP’) and Completed (Grade = ‘Some Letter Grade’) into two separate files. I then separated out Dorm data by (‘Dorm, Room’). Thus I loaded 4 data files (csv) for Students, Enrolled Courses, Completed Courses and Dorm. The commonalities amongst all the files were in the (‘ID’, ‘Firstname’, ‘Lastname’). I then mapped Students to Enrolled Courses with the ‘Enrolled’ relationship; mapped Students to Completed Courses with the ‘Completed’ relationship; mapped Students to Dorms with the ‘Housed’ relationship.

Load Students Data

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/students.csv' as students create
(c1: Students {ID: students.ID, Gender:
students.Gender, Firstname: students.Firstname,
Lastname: students.Lastname, Address: students.Address,
City: students.City, State: students.State, Zip:
students.Zip, Phone: students.Phone})
```

```
Added 6 labels, created 6 nodes, set 54 properties, returned 0 rows in 826 ms
```

Load Enrolled Courses

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/enrolledcourses.csv' as enrolled
create (e1: Enrolled {ID: enrolled.ID, Firstname:
enrolled.Firstname, Lastname: enrolled.Lastname,
Department: enrolled.Department, Coursenumber:
enrolled.Coursenumber, Grade: enrolled.Grade, Section:
enrolled.Section, Instructor: enrolled.Instructor})
```

```
Added 12 labels, created 12 nodes, set 96 properties, returned 0 rows in 206 ms
```

Load Completed Courses

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/completedcourses.csv' as
completed create (e1: Completed {ID: completed.ID,
Firstname: completed.Firstname, Lastname:
completed.Lastname, Department: completed.Department,
Coursenumber: completed.Coursenumber, Grade:
completed.Grade, Section: completed.Section,
Instructor: completed.Instructor})
```

Added 12 labels, created 12 nodes, set 96 properties, returned 0 rows in 134 ms

Load Dorm Data

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/dorms.csv' as dorms create (e1:
Dorms {ID: dorms.ID, Firstname: dorms.Firstname,
Lastname: dorms.Lastname, Dorm: dorms.Dorm, Room:
dorms.Room})
```

Added 6 labels, created 6 nodes, set 30 properties, returned 0 rows in 123 ms

Create ‘Enrolled’ Relationship between Students and Enrolled Courses

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/students.csv' as students match
(a: Students {ID: students.ID}),(b: Enrolled {ID:
students.ID}) create (a) - [r:Enrolled] -> (b)
```

Created 12 relationships, returned 0 rows in 297 ms

Create ‘Completed’ Relationship between Students and Completed Courses

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/students.csv' as students match
(a: Students {ID: students.ID}),(c: Completed {ID:
```

```
students.ID}) create (a) - [r:Completed] -> (c)
```

Created 12 relationships, returned 0 rows in 86 ms

Create 'Housed' Relationship between Students and Dorms

```
load csv with headers from 'file:/Users/MicrostrRes/
Desktop/week12QuizData/students.csv' as students match
(a: Students {ID: students.ID}),(d: Dorms {ID:
students.ID}) create (a) - [r:Housed] -> (d)
```

Created 6 relationships, returned 0 rows in 187 ms

Problem 2)

I think that using a relational data base makes sense if there isn't a tremendous amount of variation in the structure or relationships of the data. In this case, however, there seems to be a high level of variation in the data structure in that a student node can map to enrolled and/or completed courses student nodes can map to one or more dorms and one or more rooms. Each particular sub-structure in the data can change a lot if we have a lot of students and lots of courses. I would say that a relational data base would be fine if there were less dimensionality to each data structure surrounding student nodes since queries would probably operate faster on a rigid group of tables. In the graph db you have to find the node in query and follow the path outwards- however, this may be perfect for a dataset that has many different paths going into and out of each particular node. I would say to answer question #2 that Neo4j graph data base is better suited to the type of data set presented in this Quiz primarily because of the variation in relationships between the nodes.

Problem 3)

Query to Find All of Richard Kowlaski's Roomates (Note that Richard Lives in Dorm Harmon Room 301).

```
match(n:Dorms{Dorm: "Harmon",Room: "301"}) return n
```

ID	19457
Firstname	Richard
Lastname	Kowalski
Dorm	Harmon
Room	301

ID	10414
Firstname	Franklin
Lastname	Kerry
Dorm	Harmon
Room	301

Problem 4)

Modify Richard Kowlaski's grade in Math Section 12136 from IP (Enrolled) to a B (Completed). First find Richard Kowlaski's classes:

```
match(n:Enrolled{Firstname: "Richard"}) return n
```

ID	19457
Firstname	Richard
Lastname	Kowalski
Department	Mathematics
Section	12136
Grade	IP
Coursenumber	120
Instructor	Judge

ID	19457
Firstname	Richard
Lastname	Kowalski
Department	History
Section	76980
Grade	IP
Coursenumber	301
Instructor	Jones

Now Modify the Enrolled Math Grade of ‘IP’ in Section ‘12136’ to a ‘B’

```
match (n:Enrolled
{Firstname:'Richard',Section:'12136',Grade: 'IP'}) set
n.Grade = 'B' return
```

ID	19457
Firstname	Richard
Lastname	Kowalski
Department	Mathematics
Section	12136
Coursenumber	120
Instructor	Judge
Grade	B

Problem 5)

I think using Instructor as a node rather than an attribute makes a lot of sense. There are more paths that flow out of instructors to students than from students to instructors. For example,

```
match(n:Enrolled{Instructor: "Willis"}) return (n)
match(n:Enrolled{Instructor: "Willis"}) return count(n)
```

count(n)
5

There are 5 enrolled students that flow out of Professor Willis and there are 3 students that have completed courses.

```
match(n:Enrolled{Instructor: "Willis"}) return count(n)
```

count(n)
3

As such there are 8 students that flow out of Professor Willis alone. From those students would flow out dorms and students that flow to the same dorm to the same room are roommates. Students that flow from a professor into the same enrolled or completed classes are enrolled classmates struggling to complete the course or completed classmates that can celebrate together.