Peter Tadrous

CUS 725 - Advanced Database Systems

Week 12 - Homework 10

Generating the data

I decided to create all of my data in a csv (using excel for lookups, as explained below) so that loading the data would be simplified into a few statements.

To create the courses.csv, I used used 7 academic building names from *stjohns.edu*, and I generated 20 random integers between 1-7 using *random.org* to correspond to those building names, and used an excel vlookup to create the list of buildings for 20 courses. I did the same for a list of 20 room numbers between 100-300 and joined the buildings and room numbers to a single string. I manually put in course names from whatever I could think, and used *random.org* for the course numbers. The max students was also randomly generated between 15-30.

	A	B	C	D
1	courseNumber	courseName	courseLocation	maxStudents
2	BUS3017	International Business Operations	Sun Yat Sen Memorial Hall, Room 190	30
3	CUS1241	Intro to Java	Sullivan Hall, Room 246	19
4	SOC1149	Intro to Sociology	Sun Yat Sen Memorial Hall, Room 215	21
5	THE1033	Intro to Theology	St. Augustine Hall, Room 161	18

To create the students.csv, I used *name-generator.org.uk* to generate 50 random students and *random.*org to generate random majors (out of the top 10 majors from a *bestcolleges.com* blog, using a vlookup as with course building numbers).

	A	B	C	D
1	studentID	studentFirstName	studentLastName	studentMajor
2	1001	Dane	Stott	Psychology
3	1002	Elicia	Villanueva	Biology
4	1003	Amelia	Timms	Computer Science
5	1004	Sapphire	Snyder	Business
6	1005	Steven	Weston	Health Professions
7	1006	Wren	Millington	Engineering

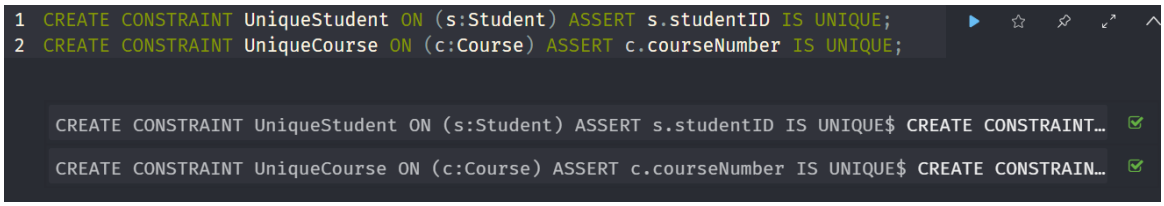
To create registrations.csv, I used random.org to generate the list of 200 studentIDs between 1001-1050 (my autoincremented studentID), as well as 200 numbers between 1-20 for course numbers (again, retrieved via a vlookup on courses). I had to remove duplicate relationships and registrations that would exceed the maximum for a course before adding the date registered, which left me with 180 unique relationships. I then used *random.org* to generate 180 random dates in ISO format. I did some manual cleanup of the registrations to try and keep students registered in courses that would be reasonable for their major but I didn't pay too much attention to this since it's random data.

	A	B	C	D
1	studentID	courseNumber	dateRegistered	
2	1026	THE1033	2019-10-22	
3	1031	BIO2231	2019-11-07	
4	1040	CUS2117	2019-12-19	
5	1025	ART2028	2020-01-21	
6	1034	COM2145	2020-03-13	

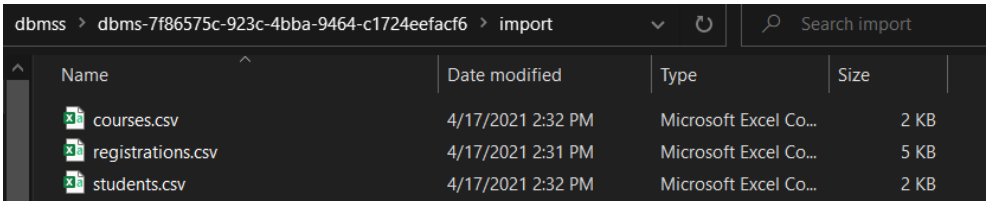
Loading the data

The first step is to create the constraints on student id and course number so that they are unique, using the following statements:

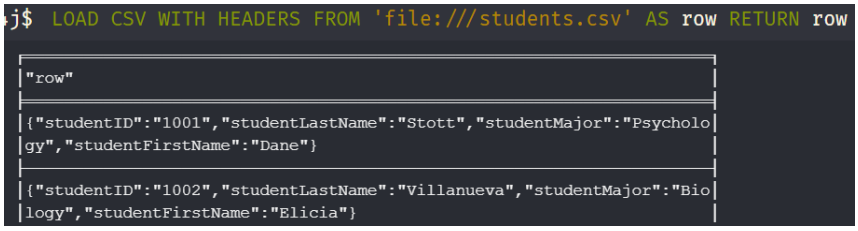
```
CREATE CONSTRAINT UniqueStudent ON (s:Student) ASSERT s.studentID IS UNIQUE;
CREATE CONSTRAINT UniqueCourse ON (c:Course) ASSERT c.courseNumber IS UNIQUE;
```



Then I added the csv files to the database's import folder for my load statements.

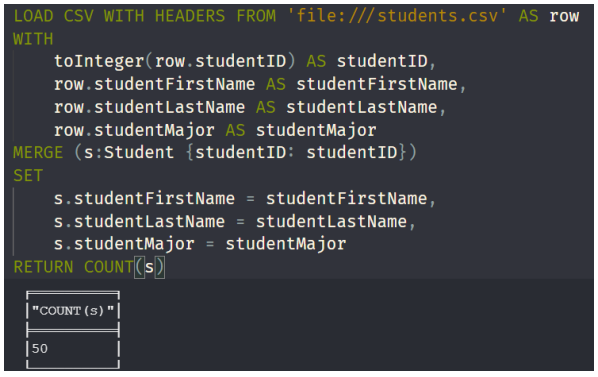


I want to make sure I can access the data correctly so I will just return the row for now. We can see that the data is loading in just fine:



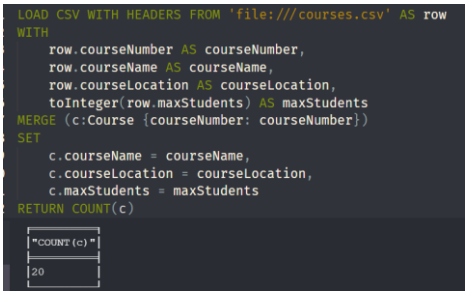
Now I can add the 50 students to the database using **MERGE** on the studentID:

```
LOAD CSV WITH HEADERS FROM 'file:///students.csv' AS row WITH
toInteger(row.studentID) AS studentID, row.studentFirstName AS
studentFirstName, row.studentLastName AS studentLastName, row.studentMajor AS
studentMajor MERGE (s:Student {studentID: studentID}) SET s.studentFirstName =
studentFirstName, s.studentLastName = studentLastName, s.studentMajor =
studentMajor RETURN COUNT(s)
```



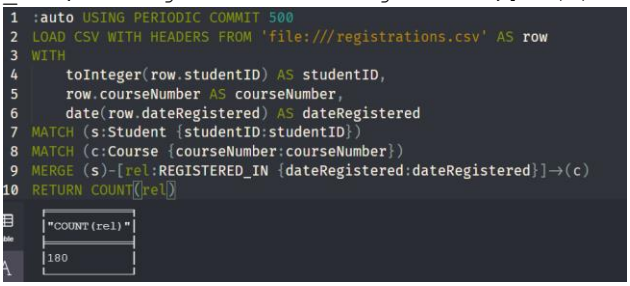
And I can add the 20 courses using **MERGE** again on the courseNumber:

```
LOAD CSV WITH HEADERS FROM 'file:///courses.csv' AS row WITH row.courseNumber
AS courseNumber,  row.courseName AS courseName, row.courseLocation AS
courseLocation,  toInteger(row.maxStudents) AS maxStudents MERGE (c:Course
{courseNumber: courseNumber}) SET c.courseName = courseName, c.courseLocation =
courseLocation, c.maxStudents = maxStudents RETURN COUNT(c)
```

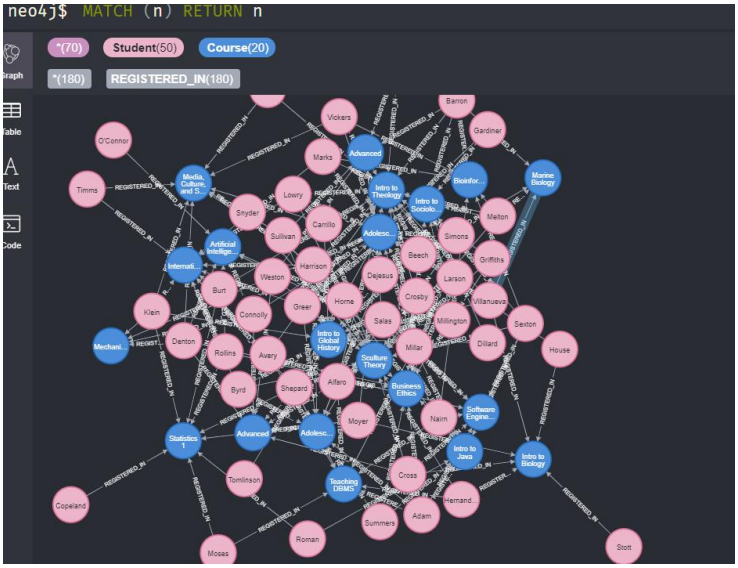


Lastly, I can add the 180 relationships, matching on studentID and courseNumber:

```
:auto USING PERIODIC COMMIT 500
LOAD CSV WITH HEADERS FROM 'file:///registrations.csv' AS row WITH
toInteger(row.studentID) AS studentID, row.courseNumber AS courseNumber,
date(row.dateRegistered) AS dateRegistered MATCH (s:Student
{studentID:studentID}) MATCH (c:Course {courseNumber:courseNumber}) MERGE (s)-
[rel:REGISTERED_IN {dateRegistered:dateRegistered}]->(c) RETURN COUNT(rel)
```



We can view the entire graph by using: `MATCH (n) RETURN n`

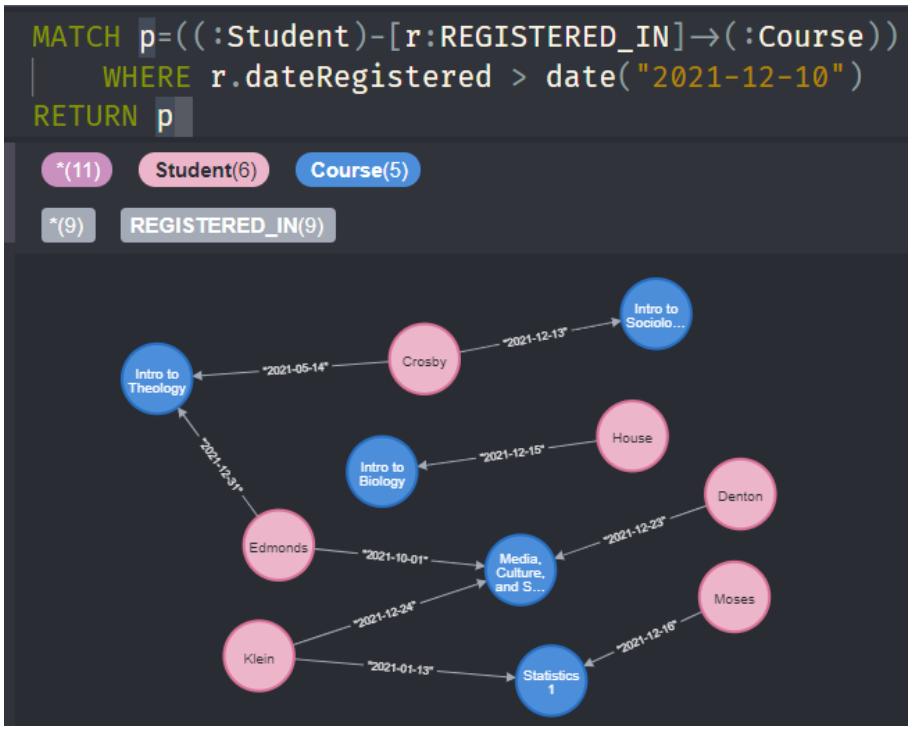


Queries

1. List the students in a particular course. (Used MTH1027)
    a. `MATCH (students:Student)-[:REGISTERED_IN]->(:Course {courseNumber:"MTH1027"}) RETURN students`

```
4j$ MATCH (students:Student)-[:REGISTERED_IN]→(:Course
    {courseNumber:"MTH1027"}) RETURN students
╒═══════════════════════════════════════════════════════════════════════════╕
│"students"                                                                 │
╞═══════════════════════════════════════════════════════════════════════════╡
│{"studentLastName":"Copeland","studentID":1015,"studentMajor":"Health      │
│Professions","studentFirstName":"Finnlay"}                                 │
├───────────────────────────────────────────────────────────────────────────┤
│{"studentLastName":"Roman","studentID":1047,"studentMajor":"Computer S     │
│cience","studentFirstName":"Milton"}                                       │
├───────────────────────────────────────────────────────────────────────────┤
│{"studentLastName":"Shepard","studentID":1049,"studentMajor":"Educatio     │
```

2. List the students, registration links and courses where the student registered after a certain date. (Used 2021-12-10, and set the caption of the relationship to **dateRegistered**)
    a. `MATCH p=((:Student)-[r:REGISTERED_IN]->(:Course)) WHERE r.dateRegistered > date("2021-12-10") RETURN p`



3. Return the number of students registered in a particular course. (Used CUS2129)
    a. `MATCH (s:Student)-[:REGISTERED_IN]->(:Course {courseNumber:"CUS2129"}) RETURN count(s) as numStudents`

```
MATCH (s:Student)-[:REGISTERED_IN]→(:Course
{courseNumber:"CUS2129"}) RETURN count(s) as numStudents
numStudents
7
```