# Week 12 Quiz Solutions

1. I used R to clean up the data to load it into Neo4j and created separate csv files for the nodes and the relationships to be loaded into Neo4j.

Provided below is the script for data clean up in R:

require(dplyr)

df<- read.csv(file.choose(), header=TRUE)

df2<- read.csv(file.choose(), header=TRUE)

studentdf<-select(df2, ID:TelephoneNumber)

studentdf

write.csv(studentdf, file = "Student.csv")

dormitorydf<-unique(select(df2, Dormitory))

write.csv(dormitorydf, file = "Dormitory.csv")

coursedf<-unique(select(df, CourseDept:CourseName))

write.csv(coursedf, file = "Course.csv")

EnrolledCourse<-filter(df, Grade=='IP')

write.csv(EnrolledCourse, file = "EnrolledCourse.csv")

CompletedCourse<-filter(df, Grade!='IP')

write.csv(CompletedCourse, file = "CompletedCourse.csv")

From the csv files generated above I loaded the data into Neo4j to create nodes and relationships:

load csv with headers from "file:C:\\data\\Student.csv" as students create (s1:Student {ID:students.ID, Gender:students.Gender, GivenName:students.GivenName, Surname:students.Surname,StreetAddress:students.StreetAddress, City:students.City, State:students.State, ZipCode:students.ZipCode, Telephone:students.TelephoneNumber})

load csv with headers from "file:C:\\data\\Course.csv" as course create (c:Course {CourseDept:course.CourseDept, CourseNumber:course.CourseNumber, CourseName:course.CourseName})

load csv with headers from "file:C:\\data\\Dormitory.csv" as dorm create (d:Dormitory {Dormitory:dorm.Dormitory})

load csv with headers from "file:C:\\data\\EnrolledCourse.csv" as enrolled match (s:Student {ID:enrolled.ID}), (c:Course{CourseNumber:enrolled.CourseNumber}) create (s)-[r:Enrolled{Grade:enrolled.Grade, Section:enrolled.Section, Instructor:enrolled.Instructor}]->(c)

load csv with headers from "file:C:\\data\\CompletedCourse.csv" as completed match (s:Student {ID:completed.ID}), (c:Course{CourseNumber:completed.CourseNumber}) create (s)-[r:Completed{Grade:completed.Grade, Section:completed.Section, Instructor:completed.Instructor}]->(c)

load csv with headers from "file:C:\\data\\week-12-quiz-data-students-and-housing.csv" as housing match (s:Student {ID:housing.ID}), (d:Dormitory{Dormitory:housing.Dormitory}) create (s)-[r:Housed{Room:housing.Room}]->(d)

match(n) return(n)

1. My opinion will be in favor of graph database over SQL database.
   1. To implement this in SQL database we have to normalize the data to reduce redundancy but this is not required in Graph database.
   2. In SQL database we can divide the data into three tables where Student, Course and Housing and in that case Student table will be the parent table and the Course table and the Housing table will be the child tables. The Student table will hold the foreign key to those child tables. Here the relationship between Student and the Course or Housing will come as an attribute to those tables. But it is very logical in graph database that they are not actually the properties of the entities but holds relationship between them.
2. MATCH (s:Student {GivenName: 'Richard', Surname: 'Kowalski'}) - [h1:Housed] -> (m) <- [h2:Housed] - (d:Student) WHERE h1.Room = h2.Room RETURN d.GivenName, d.Surname
3. MATCH (n:Student {GivenName:"Richard", Surname:"Kowalski"})-[r:Enrolled{Section:"12136"}]->(c:Course {CourseDept:"Mathematics", CourseNumber:"120"})

CREATE (n)-[r2:Completed]->(c)

SET r2=r

MATCH (n:Student {GivenName:"Richard", Surname:"Kowalski"})-[r:Enrolled{Section:"12136"}]->(c:Course {CourseDept:"Mathematics", CourseNumber:"120"})

DELETE r

MATCH (n:Student {GivenName:"Richard", Surname:"Kowalski"})-[r:Completed{Section:"12136"}]->(c:Course {CourseDept:"Mathematics", CourseNumber:"120"})

SET r.Grade="B"