**Lab Exercise 1**

1.1 Exercise:

In this exercise you are tasked to create the graph illustrated below by just using Cypher. You are free to use as many Cypher commands as you like as long as they achieve the same result as the graph provided. Note that there are multiple ways to create the following graph, don’t be concerned if someone else used a different way.

# **Creation of Nodes ~**

create (u:University {name:"UNIVERSITY"})

create (m:Module {name:"COMP1831"})

create (m:Module {name:"COMP1830"})

create (s:Student {name:"Rasvan"})

create (s:Student {name:"Rajesh"})

create (s:Student {name:"George"})

create (s:Student {name:"Peter"})

create (s:Student {name:"Olivia"})

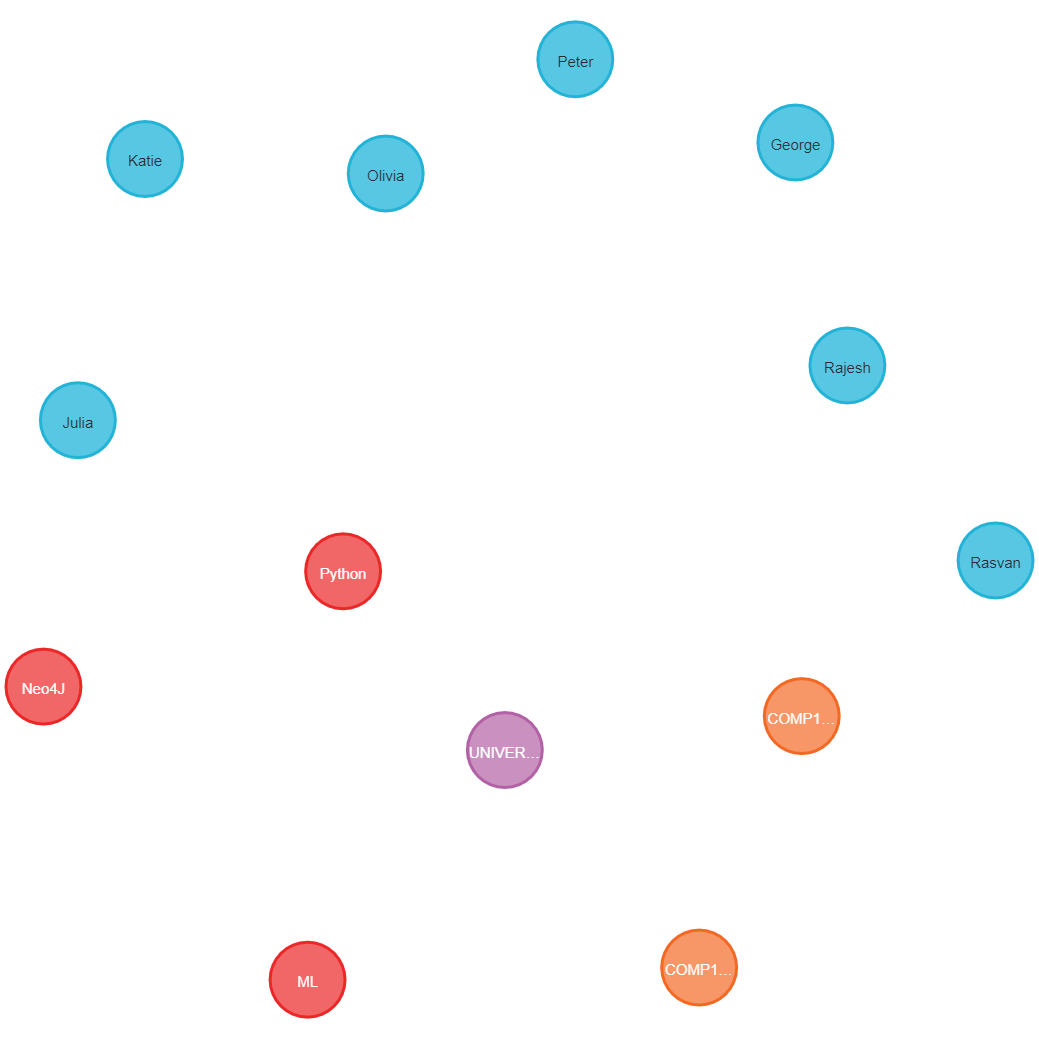
create (s:Student {name:"Katie"})

create (s:Student {name:"Julia"})

create (t:Technology {name:"Neo4J"})

create (t:Technology {name:"Python"})

create (t:Technology {name:"ML"})



**NODES**

# Matching nodes & Associating properties~

Match

(u:University),

(s:Student)

where u.name ="UNIVERSITY" AND s.name="Julia"

Create (u)-[studies:STUDIES\_IN]->(s)

Return studies

Match

(u:University),

(s:Student)

where u.name ="UNIVERSITY" AND s.name="Olivia"

Create (u)-[studies:STUDIES\_IN]->(s)

Return studies

Match

(u:University),

(s:Student)

where u.name ="UNIVERSITY" AND s.name="Peter"

Create (u)-[studies:STUDIES\_IN]->(s)

Return studies

Match

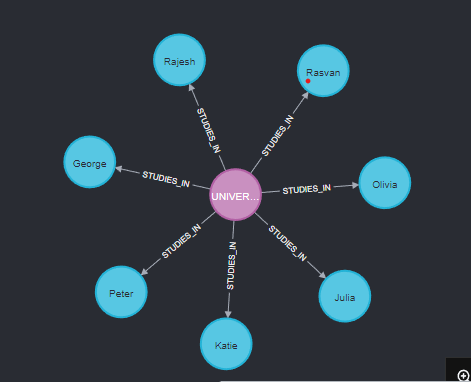
(u:University),

(s:Student)

where u.name ="UNIVERSITY" AND s.name="George"

Create (u)-[studies:STUDIES\_IN]->(s)

Return studies



Match

(s:Student),

(m:Module)

where s.name ="Julia" AND m.name="COMP1831"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="Olivia" AND m.name="COMP1831"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="Peter" AND m.name="COMP1831"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="George" AND m.name="COMP1831"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="George" AND m.name="COMP1830"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="Rasvan" AND m.name="COMP1830"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="Rajesh" AND m.name="COMP1830"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(m:Module)

where s.name ="Katie" AND m.name="COMP1831"

Create (s)-[attends:ATTENDS]->(m)

Return attends

Match

(s:Student),

(t:Technology)

where s.name ="Peter" AND t.name="ML"

Create (s)-[works:WORKED\_WITH]->(t)

Return works

Match

(s:Student),

(t:Technology)

where s.name ="George" AND t.name="Python"

Create (s)-[works:WORKED\_WITH]->(t)

Return works

Match

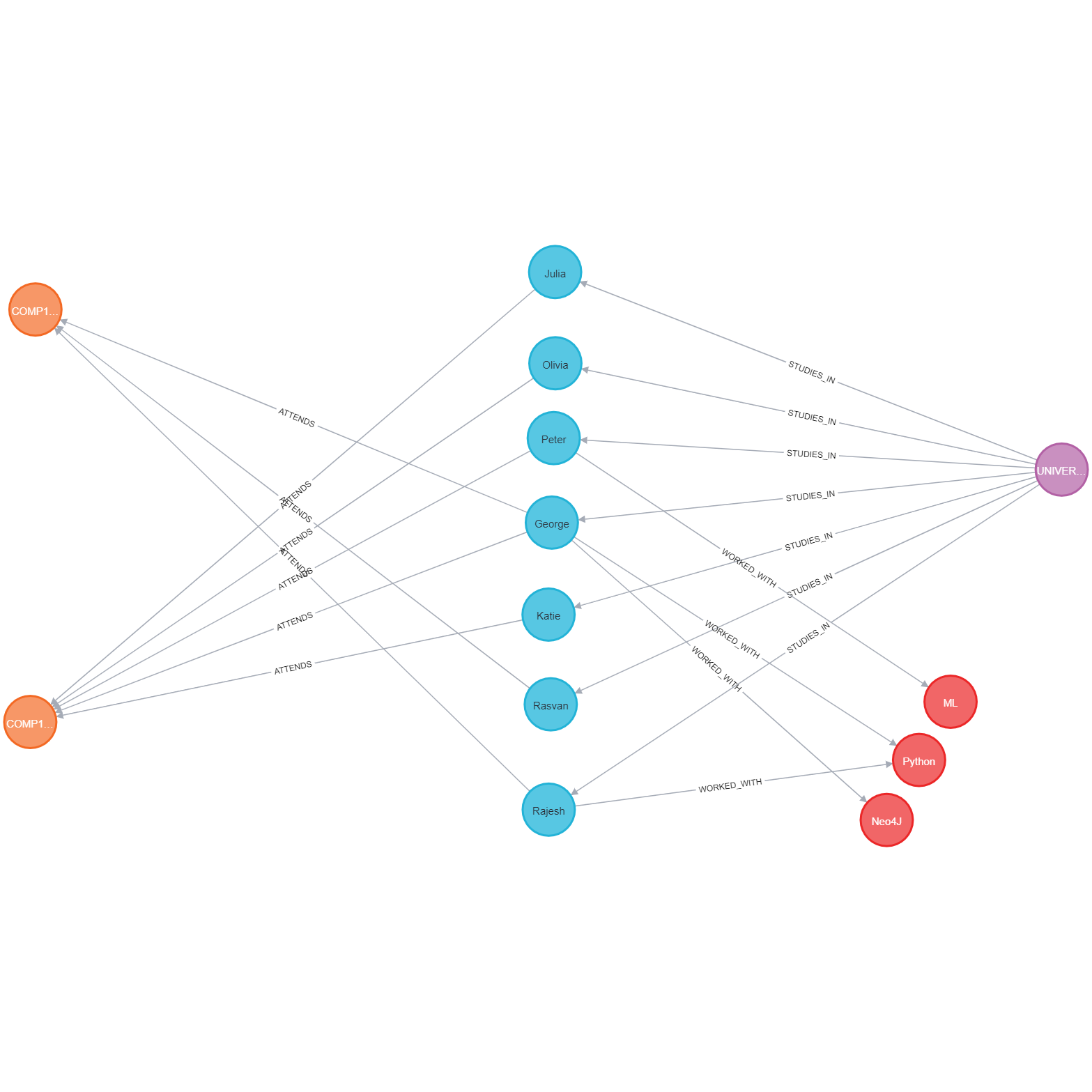
(s:Student),

(t:Technology)

where s.name ="George" AND t.name="Neo4J"

Create (s)-[works:WORKED\_WITH]->(t)

Return works



1.2 Exercise:

Perform a Cypher query to find a student who attends “COMP1831” and has worked with

both “Neo4J” and “Python” before.

2.1 Exercise:

Import the following CSV data files (persons2.csv, loans.csv and status.csv) into the

following graph data model.

create constraint personIdConstraint FOR (person:Person) REQUIRE person.id IS UNIQUE

create constraint loanIdConstraint FOR (loan:Loan) REQUIRE loan.id IS UNIQUE

LOAD CSV WITH HEADERS FROM "https://gist.githubusercontent.com/apogiatzis/237146155b85be73e25d775e487f4d8c/raw/7a440b1f4b7885ea425d5db0ce6aae041b8ea42a/persons2.csv" As csvLine create (person:Person{id:toInteger(csvLine.personId),firstName:csvLine.firstName,lastName:csvLine.lastName})

LOAD CSV WITH HEADERS FROM "https://gist.githubusercontent.com/apogiatzis/93927b266820ce9696b94055e896b1f9/raw/fdfdd19194b2afc1f78b01e4ddd783826b676878/loans.csv" As csvLine merge (bank:Bank{name:csvLine.bank}) create(loan:Loan{id:toInteger(csvLine.loanId),amount:csvLine.amount})

create (loan)-[:FROM]->(bank)

LOAD CSV WITH HEADERS FROM "https://gist.githubusercontent.com/apogiatzis/dbc8f04359311257cd84e216ae8fedc4/raw/19fa315206573cab427d8b13e641c403a45852c6/status.csv" As csvLine match (person:Person{id:toInteger(csvLine.personId)}),(loan:Loan{id:toInteger(csvLine.loanId)}) create (person)-[:RECEIVED{status:csvLine.status}]->(loan)

match (n) return \*

