Liam J
Jake M

```
//Item 1. List Names of students and their mentors.
MATCH (sp:Person)<-[:S2P]-(s)<-[:Mentors]-(m)-[M2P]-(mp)
WHERE exists(mp.Name) AND exists(sp.Name)
RETURN sp.Name as Student_Name, mp.Name as Mentor_Name

//Item 2. For classification "Sophomore", list the classification (Sophomore) and the
average GPA of students with that classification.
MATCH (s:Student)
WHERE s.Classification='Sophomore'
RETURN s.Classification as Classification, AVG(s.GPA) as Average_GPA

//Item 3. For each enrolled student, list his/her name and his/her mentor's name.
MATCH (stu)<-[:S2P]-(s2)-[:Taking]->(e)<-[:Taking]-(s1:Student)<-[:Mentors]-(m)-[:F2P]-
>(ment)
RETURN stu.Name as Student_Name, ment.Name as Mentor_Name

//Item 4. List the Names of students who have at least one A in courses they are enrolled
in. (Do not consider A- as an A.)
MATCH (a)<-[:S2P]-(s:Student)-[t:Taking]->(e)
WHERE t.Grade ='A'
RETURN DISTINCT a.Name as Student_Name

//Item 5. List the course codes of courses that are prerequisites for the course 'CS311'.
MATCH (pr:Course)-[:PreReq]->(c)
WHERE c.CourseCode='CS311'
RETURN pr.CourseCode as Prerequisites

//Item 6. List the Name, and Rank of faculty who have a salary of 100K or higher.
MATCH (f:Faculty)-[:F2P]->(p)
WHERE f.Salary>=100000
RETURN p.Name as Name, f.Rank as Rank

//Item 7. For each faculty, list his / her name, and classification and GPA of students that
the faculty is mentoring.
MATCH (s)<-[:Mentors]-(f:Faculty)-[:F2P]->(p)
RETURN p.Name as Name, s.Classification as Student_Classification, s.GPA as Student_GPA

//Item 8. For every offering, list course name, section, instructors' name, and enrollment.
MATCH (e)<-[:Covers]-(o)-[tr:Taking]-(s)-[:Mentors]-(f)-[:F2P]-(p)
RETURN e.CourseName as Course, collect(DISTINCT o.Section) as Sections, collect(DISTINCT
p.Name) as Instructors_Name, collect(DISTINCT tr.EnrollmentID) as Enrollment
```