```
//Most-connected organizations
match (o:Organization)-[r]-()
return o.name, count(*), collect(distinct type(r)) as types
order by count(*) desc
limit 10

//The second-degree network of Kushner
match network=(:Person {name:"JARED KUSHNER"})-[*..2]-()
return network

//Trump's possible connections to Putin
match path=allShortestPaths((dt:Person {name:"DONALD J. TRUMP"})-[*]-(vp:Person {name:"VLADIMIR PUTIN"}))
return path

//Clustering Coefficient
match path1=(p0:Person)-[:Related_to]-(dt:Person {name:"DONALD J. TRUMP"})-[:Related_to]-(p2:Person)
where not ((p0)-[:Related_to]-(p2)) and id(p0)<id(p2)
with count(path1) as m
match path2=(p0:Person)-[:Related_to]-(dt:Person {name:"DONALD J. TRUMP"})-[:Related_to]-(p2:Person)
where id(p0)<id(p2)
return 1.0*m/count(path2) as Cluster_Coef

//Degree Centrality Using GDS
call gds.graph.create("projected_graph1", "Person", "Related_to");
CALL gds.degree.stream('projected_graph1')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name AS nodeName, score as DegreeCentralityScore
ORDER BY DegreeCentralityScore DESC;

//PageRank
CALL gds.pageRank.stream('projected_graph1')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name AS name, score as PageRankScore
ORDER BY PageRankScore DESC

//Article PageRank
call gds.graph.create("projected_graph2", "Organization", "Connected_to");

CALL gds.articleRank.stream('projected_graph2')
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name AS name, score as APRScore
ORDER BY APRScore DESC

//Overlapping Similarity
MATCH (n1:Person)-[:Involved_with]->(org:Organization)
WITH {item:id(n1), categories: collect(id(org))} AS userData
WITH collect(userData) AS data
CALL gds.alpha.similarity.overlap.stream({data: data})
YIELD item1, item2, count1, count2, intersection, similarity
RETURN gds.util.asNode(item1).name AS from, gds.util.asNode(item2).name AS to,
       count1, count2, intersection, similarity
ORDER BY similarity DESC
```