1)

CALL gds.graph.create('WineSimilarity', ['Wine', 'Region1', 'Variety', 'Points', 'Price'], '\*')

CALL gds.nodeSimilarity.stream(

'WineSimilarity',

{

topN: 10

}

)

YIELD node1, node2, similarity

RETURN gds.util.asNode(node1).name AS Wine1, gds.util.asNode(node2).name AS Wine2, similarity

2)

CALL gds.nodeSimilarity.stream(

1. 'WineSimilarity',
2. {
3. degreeCutoff: 2,
4. topN: 10,
5. topK: 1
6. }
7. )
8. YIELD node1, node2, similarity
9. RETURN gds.util.asNode(node1).name AS Wine1, gds.util.asNode(node2).name AS Wine2, similarity
10. ORDER BY similarity DESC

3) CALL gds.graph.create(

'Com-graph', {

Wine: { label: 'Wine'},

Points: { label: 'Points'},

Winery: { label: 'Winery' },

Variety: { label: 'Variety' },

Price: { label: 'Price' }

},

'\*'

)

YIELD graphName, nodeCount, relationshipCount;

CALL gds.louvain.stream(

'Com-graph',

{

includeIntermediateCommunities: true

}

)

YIELD nodeId, communityId, intermediateCommunityIds

RETURN communityId, COUNT(DISTINCT nodeId) AS Members, intermediateCommunityIds