ToDo:

1.Create „community“ nodes: consist of combinations of keywords

// Create (or match) the ResearchCommunity node and link it to its keywords

MERGE (rc:ResearchCommunity {name: "Database Community"})

WITH rc

MATCH (k:Keyword)

WHERE k.name IN [

"data management",

"indexing",

"data modeling",

"big data",

"data processing",

"data storage",

"data querying"

]

MERGE (rc)-[:HAS\_KEYWORD]->(k)

RETURN rc, collect(k.name) AS keywords;

2. Journal/Workshop/Conference 🡪 paper -> keywords to find journal/workshop/conference interest, 90% threshold

MATCH (rc:ResearchCommunity {name:"Database Community"})

// Get the list of community keywords from connected Keyword nodes

MATCH (rc)-[:HAS\_KEYWORD]->(k:Keyword)

WITH rc, collect(k.name) AS dbKeywords

MATCH (p:Paper)-[:PUBLISHED\_IN\_VENUE]->(v:PublicationVenue)

WITH v, p, dbKeywords

// Collect all keywords attached to the paper

OPTIONAL MATCH (p)-[:IN\_FIELD]->(f:Keyword)

WITH v, p, dbKeywords, collect(f.name) AS paperKeywords

// For each paper, check if it has at least one matching keyword

WITH v, count(p) AS totalPapers,

sum( CASE WHEN any(keyword IN paperKeywords WHERE keyword IN dbKeywords) THEN 1 ELSE 0 END ) AS dbPapers

WHERE totalPapers > 0 AND (dbPapers \* 1.0 / totalPapers) >= 0.9

MERGE (v)-[:RELATED\_TO {ratio: dbPapers\*1.0/totalPapers}]->(rc)

RETURN v.name AS Venue, totalPapers, dbPapers;

3. From those journals etc. find the top 100 papers with citations

MATCH (rc:ResearchCommunity {name:"Database Community"})

MATCH (v:PublicationVenue)-[:RELATED\_TO]->(rc)

MATCH (p:Paper)-[:PUBLISHED\_IN\_VENUE]->(v)

WITH rc, p

// Retrieve the community keywords from the ResearchCommunity node

MATCH (rc)-[:HAS\_KEYWORD]->(k:Keyword)

WITH rc, p, collect(k.name) AS dbKeywords

// Count citations from papers that are related to the community (i.e., connected to one of the community's keywords)

OPTIONAL MATCH (p)<-[:CITES]-(citing:Paper)

OPTIONAL MATCH (citing)-[:IN\_FIELD]->(f:Keyword)

WHERE f.name IN dbKeywords

WITH p, count(DISTINCT citing) AS citations, rc

ORDER BY citations DESC

LIMIT 100

// Mark these as top papers and connect them to the research community

SET p :TopPaper, p.topCitations = citations

MERGE (rc)-[:HAS\_TOP\_PAPER]->(p)

RETURN p.title AS Title, citations;

4. If an author has one paper from those top 100, he is a good match. If he has 2, then he is a guru

MATCH (rc:ResearchCommunity {name:"Database Community"})

MATCH (p:TopPaper)

MATCH (p)-[:WRITTEN\_BY]->(a:Author)

MERGE (a)-[:POTENTIAL\_REVIEWER]->(rc)

WITH a, count(p) AS topPaperCount, rc

WHERE topPaperCount >= 2

MERGE (a)-[:GURU {topPapers: topPaperCount}]->(rc)

RETURN a.name AS Author, topPaperCount;