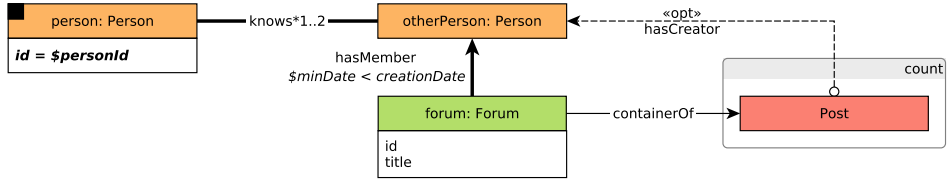


## Interactive / complex / 5

IC 1	query	Interactive / complex / 5			
IC 2	title	New groups			
IC 3	pattern	 <pre> graph LR     P1[person: Person id = \$personId] -- knows*1..2 --&gt; P2[otherPerson: Person]     P2 -- hasMember \$minDate &lt; creationDate --&gt; F[forum: Forum id title]     F -- containerOf --&gt; Post[Post]     Post -- «opt» hasCreator --&gt; P2     Post -- count --&gt; Count[count]           </pre>			
IC 5	desc.	<p>Given a start Person, find the Forums which that Person's friends and friends of friends (excluding start Person) became Members of after a given date. For each Forum find the number of Posts that were created by any of these Persons. For each Forum and consider only those Persons which joined that particular Forum after the given date (minDate).</p>			
IC 14	params	1	personId	ID	
		2	minDate	Date	
	result	1	forum.title	Long String	R
		2	postCount	32-bit Integer	A
		Number of Posts made in forum that were created by friends			
	sort	1	postCount	↓	
		2	forum.id	↑	
	limit	20			
	CPs	2.3, 3.3, 8.2, 8.5			
	relevance	<p>This query looks for paths of length two and three, starting from a given Person, moving to friends and friends of friends, and then getting the Forums they are members of. Besides testing the ability of the query optimizer to select the proper join operator, it rewards the usage of indexes, but their accesses will be presumably scattered due to the two/three-hop search space of the query, leading to unpredictable and scattered index accesses. Having efficient implementations of such indexes will be highly beneficial.</p>			