

Interactive / complex / 6

IC 1	query	Interactive / complex / 6			
IC 2	title	Tag co-occurrence			
IC 3	pattern	 <pre> graph TD person[person: Person] -- knows*1..2 --> otherPerson[otherPerson: Person] tag[tag: Tag] -- hasTag --> Post[Post] otherTag[otherTag: Tag] -- hasTag --> Post Post -- hasCreator --> otherPerson subgraph count Post end </pre> <p>The diagram illustrates the query pattern. It starts with a person: Person entity (orange box) with a variable <code>id = \$personId</code>. This person is connected via a <code>knows*1..2</code> relationship to an otherPerson: Person entity (orange box). The otherPerson is connected via a <code>hasCreator</code> relationship to a Post entity (red box). The Post entity is also connected via <code>hasTag</code> relationships to two Tag entities: tag: Tag (pink box) and otherTag: Tag (pink box). The tag entity has a variable <code>name = \$tagName</code>. The otherTag entity has a variable <code>name ≠ \$tagName</code>. A <code>count</code> aggregation is applied to the Post entity.</p>			
IC 6					
IC 7					
IC 8					
IC 9					
IC 10					
IC 11					
IC 12					
IC 13	desc.	Given a start Person and some Tag, find the other Tags that occur together with this Tag on Posts that were created by start Person's friends and friends of friends (excluding start Person). Return top 10 Tags, and the count of Posts that were created by these Persons, which contain both this Tag and the given Tag.			
IC 14	params	1	personId	ID	
		2	tagName	Long String	
	result	1	otherTag.name	Long String	R
		2	postCount	32-bit Integer	A
	sort	1	postCount	↓	
		2	otherTag.name	↑	
	limit	10			
	CPs	5.1, 8.2			
	relevance	This query looks for paths of lengths three or four, starting from a given Person, moving to friends or friends of friends, then to Posts and finally ending at a given Tag.			