

Interactive / complex / 10

IC 1
IC 2
IC 3
IC 4
IC 5
IC 6
IC 7
IC 8
IC 9
IC 10
IC 11
IC 12
IC 13
IC 14

query	Interactive / complex / 10				
title	Friend recommendation				
pattern	<p>The diagram illustrates the query pattern. At the top, a rootPerson: Person box with attribute <code>id = \$id</code> is connected via a <code>knows*2..2</code> relationship to a person: Person box. The person: Person box has attributes <code>birthday cond's</code>, <code>id</code>, <code>firstName</code>, <code>lastName</code>, and <code>gender</code>. It is connected via an <code>isLocatedIn</code> relationship to a City box with attribute <code>name</code>. Below this, two dashed boxes represent sub-queries. The common box shows a rootPerson: Person connected to a Tag via <code>hasInterest</code>, and a person: Person connected to a Post via <code>hasCreator</code>. The Post is connected to a Tag via <code>hasTag</code>. A <code>count</code> relationship is shown between the Post and the Tag. The uncommon box shows a similar structure, but the <code>hasInterest</code> relationship is dashed and red, indicating it is not present in the common case.</p>				
desc.	<p>Given a start Person with id <code>personId</code>, find that Person’s friends of friends (<code>person</code>) – excluding the start Person and his/her immediate friends –, who were born on or after the 21st of a given month (in any year) and before the 22nd of the following month. Calculate the similarity between each person and the start Person, where <code>commonInterestScore</code> is defined as follows:</p> <ul style="list-style-type: none">• <code>common</code> = number of Posts created by <code>person</code>, such that the Post has a Tag that the start Person, is interested in• <code>uncommon</code> = number of Posts created by <code>person</code>, such that the Post has no Tag that the start Person, is interested in• <code>commonInterestScore</code> = <code>common</code> - <code>uncommon</code>				
params	<div><div>1</div><div>Person.id</div><div>ID</div></div>	<div>personId</div>			
	<div><div>2</div><div>month</div><div>32-bit Integer</div></div>	<div>month – Between 1 and 12. Implementations may also pass the next month as an additional <code>nextMonth</code> parameter</div>			
result	<div><div>1</div><div>Person.id</div><div>ID</div></div>	<div>R</div>	<div>personId</div>		
	<div><div>2</div><div>Person.firstName</div><div>String</div></div>	<div>R</div>	<div>personFirstName</div>		
	<div><div>3</div><div>Person.lastName</div><div>String</div></div>	<div>R</div>	<div>personLastName</div>		
	<div><div>4</div><div>commonInterestScore</div><div>32-bit Integer</div></div>	<div>C</div>	<div>commonInterestScore</div>		
	<div><div>5</div><div>Person.gender</div><div>String</div></div>	<div>R</div>	<div>personGender</div>		
	<div><div>6</div><div>Person-isLocatedIn->City.name</div><div>String</div></div>	<div>R</div>	<div>personCityName</div>		
sort	<div><div>1</div><div>commonInterestScore</div><div>↓</div></div>				
	<div><div>2</div><div>Person.id</div><div>↑</div></div>				
limit	10				
CPs	2.3, 3.3, 4.1, 4.2, 5.1, 5.2, 6.1, 7.1, 8.6				
relevance	<p>This query looks for paths of length two, starting from a Person and ending at the friends of their friends. It does widely scattered graph traversal, and one expects no locality of in friends of friends, as these have been acquired over a long time and have widely scattered identifiers. The join order is simple but one must see that the anti-join for “not in my friends” is better with hash. Also the last pattern in the scalar sub-queries joining or anti-joining the Tags of the candidate’s Posts to interests of self should be by hash.</p>				