

## Interactive / complex / 1

query	Interactive / complex / 1																																																																					
title	Transitive friends with certain name																																																																					
pattern	<pre>graph LR     P1[person: Person id = \$personId] -- knows*1..3 --&gt; P2[otherPerson: Person firstName = \$firstName id lastName birthday creationDate gender browserUsed locationIP email speaks]     P2 -- «opt» workAt --&gt; C[company: Company name]     P2 -- «opt» studyAt --&gt; U[university: University name]     C -- isLocatedIn --&gt; CC[companyCountry: Country name]     U -- isLocatedIn --&gt; UC[universityCity: City name]</pre>																																																																					
desc.	Given a start Person, find Persons with a given first name (firstName) that the start Person is connected to (excluding start Person) by at most 3 steps via the knows relationships. Return Persons, including the distance (1..3), summaries of the Persons workplaces and places of study.																																																																					
params	<table><tr><td>1</td><td>personId</td><td>ID</td><td></td></tr><tr><td>2</td><td>firstName</td><td>String</td><td></td></tr></table>					1	personId	ID		2	firstName	String																																																										
1	personId	ID																																																																				
2	firstName	String																																																																				
result	<table><tr><td>1</td><td>otherPerson.id</td><td>ID</td><td>R</td><td></td></tr><tr><td>2</td><td>otherPerson.lastName</td><td>String</td><td>R</td><td></td></tr><tr><td>3</td><td>distanceFromPerson</td><td>32-bit Integer</td><td>C</td><td></td></tr><tr><td>4</td><td>otherPerson.birthday</td><td>Date</td><td>R</td><td></td></tr><tr><td>5</td><td>otherPerson.creationDate</td><td>DateTime</td><td>R</td><td></td></tr><tr><td>6</td><td>otherPerson.gender</td><td>String</td><td>R</td><td></td></tr><tr><td>7</td><td>otherPerson.browserUsed</td><td>String</td><td>R</td><td></td></tr><tr><td>8</td><td>otherPerson.locationIP</td><td>String</td><td>R</td><td></td></tr><tr><td>9</td><td>otherPerson.email</td><td>{Long String}</td><td>R</td><td></td></tr><tr><td>10</td><td>otherPerson.speaks</td><td>{String}</td><td>R</td><td></td></tr><tr><td>11</td><td>locationCity.name</td><td>String</td><td>R</td><td></td></tr><tr><td>12</td><td>universities</td><td>{&lt;String, 32-bit Integer, String&gt;}</td><td>A</td><td>{&lt;university.name, studyAt.classYear, universityCity.name&gt;}</td></tr><tr><td>13</td><td>companies</td><td>{&lt;String, 32-bit Integer, String&gt;}</td><td>A</td><td>{&lt;company.name, workAt.workFrom, companyCountry.name&gt;}</td></tr></table>					1	otherPerson.id	ID	R		2	otherPerson.lastName	String	R		3	distanceFromPerson	32-bit Integer	C		4	otherPerson.birthday	Date	R		5	otherPerson.creationDate	DateTime	R		6	otherPerson.gender	String	R		7	otherPerson.browserUsed	String	R		8	otherPerson.locationIP	String	R		9	otherPerson.email	{Long String}	R		10	otherPerson.speaks	{String}	R		11	locationCity.name	String	R		12	universities	{<String, 32-bit Integer, String>}	A	{<university.name, studyAt.classYear, universityCity.name>}	13	companies	{<String, 32-bit Integer, String>}	A	{<company.name, workAt.workFrom, companyCountry.name>}
1	otherPerson.id	ID	R																																																																			
2	otherPerson.lastName	String	R																																																																			
3	distanceFromPerson	32-bit Integer	C																																																																			
4	otherPerson.birthday	Date	R																																																																			
5	otherPerson.creationDate	DateTime	R																																																																			
6	otherPerson.gender	String	R																																																																			
7	otherPerson.browserUsed	String	R																																																																			
8	otherPerson.locationIP	String	R																																																																			
9	otherPerson.email	{Long String}	R																																																																			
10	otherPerson.speaks	{String}	R																																																																			
11	locationCity.name	String	R																																																																			
12	universities	{<String, 32-bit Integer, String>}	A	{<university.name, studyAt.classYear, universityCity.name>}																																																																		
13	companies	{<String, 32-bit Integer, String>}	A	{<company.name, workAt.workFrom, companyCountry.name>}																																																																		
sort	<table><tr><td>1</td><td>distanceFromPerson</td><td>↑</td><td></td></tr><tr><td>2</td><td>otherPerson.lastName</td><td>↑</td><td></td></tr><tr><td>3</td><td>otherPerson.id</td><td>↑</td><td></td></tr></table>					1	distanceFromPerson	↑		2	otherPerson.lastName	↑		3	otherPerson.id	↑																																																						
1	distanceFromPerson	↑																																																																				
2	otherPerson.lastName	↑																																																																				
3	otherPerson.id	↑																																																																				
limit	20																																																																					
CPs	2.1, 5.3, 8.2																																																																					
relevance	This query is a representative of a simple navigational query. It looks for paths of length 1..3 through the knows relation, starting from a given Person and ending at a Person with a given first name. It is interesting for several aspects. (1) It requires for a complex aggregation for returning the concatenation of universities, companies, languages and email information of the Person. (2) It tests the ability of the optimizer to move the evaluation of sub-queries functionally dependant on the Person, after the evaluation of the top-k. (3) Its performance is highly sensitive to properly estimating the cardinalities in each transitive path, and paying attention not to explore already visited Persons.																																																																					