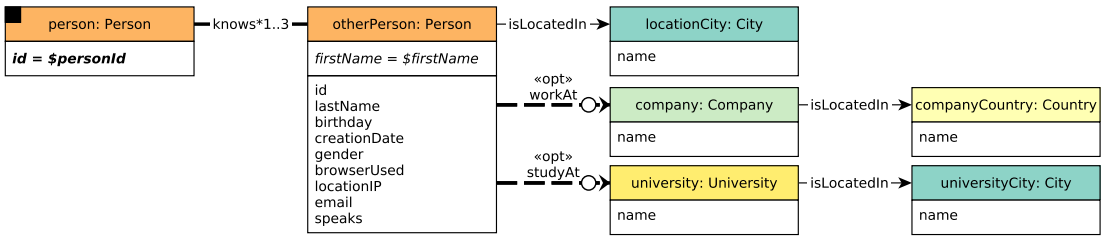
Interactive / complex / 1

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 / 1
title	Transitive friends with certain name
pattern	
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

1	personId	ID	
2	firstName	String	

result

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

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.