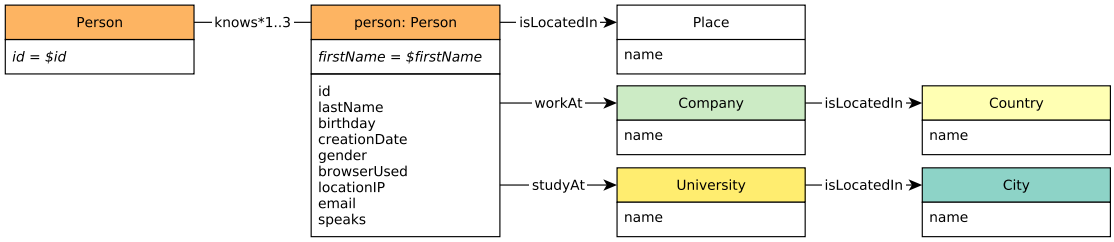
Interactive / complex / 1

query	Interactive / complex / 1
title	Friends with certain name
pattern	
desc.	Given a start Person, find Persons with a given first name that the start Person is connected to (excluding start Person) by at most 3 steps via Knows relationships. Return Persons, including summaries of the Persons workplaces and places of study.

params

1	Person.id	ID	
2	Person.firstName	String	

result

1	Person.id	ID	R	
2	Person.lastName	String	R	
3	Person.birthday	Date	R	
4	Person.creationDate	DateTime	R	
5	Person.gender	String	R	
6	Person.browserUsed	String	R	
7	Person.locationIP	String	R	
8	{Person.email}	{String}	R	
9	{Person.speaks}	{String}	R	
10	Person-isLocatedIn->Place.name	String	R	
11	{Person-studyAt->University.name, Person-studyAt->.classYear, Person-studyAt->University-isLocatedIn->City.name}	{}	R	
12	{Person-workAt->Company.name, Person-workAt->.workFrom, Person-workAt->Company-isLocatedIn->Country.name}	{}	R	

sort

1	distanceFromPerson	↑	
2	Person.lastName	↑	
3	Person.id	↑	

limit	20
CPs	1.3, 2.1, 5.3
relevance	This query is a representative of a simple navigational query. It looks for paths of length one, two or three through the knows relation, starting from a given Person and ending at a Person with a given name. It is interesting for several aspects. First, it requires for a complex aggregation, that is, returning the concatenation of universities, companies, languages and email information of the person. Second, 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. Finally, performance is highly sensitive to properly estimating the cardinalities in each transitive path, and paying attention not to explode already visited Persons.