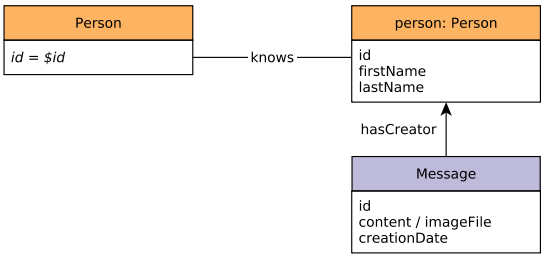
IC 1
IC 2
IC 3
IC 4
IC 5
IC 6
IC 7
IC 8
IC 9
IC 10
IC 11
IC 12
IC 13
IC 14

Interactive / complex / 2

query	Interactive / complex / 2
title	Recent messages by your friends
pattern	
desc.	Given a start Person, find (the most recent) Messages from all of that Person's friends. Only consider Messages created before the given `maxDate` (excluding that day).

params

1	`Person.id`	ID	`personId`
2	`maxDate`	Date	`maxDate`

result

1	`Message-hasCreator->Person.id`	ID	R	`personId`
2	`Message-hasCreator->Person.firstName`	String	R	`personFirstName`
3	`Message-hasCreator->Person.lastName`	String	R	`personLastName`
4	`Message.id`	ID	R	`messageId`
5	`Message.content or Post.imageFile`	String	R	`messageContent`
6	`Message.creationDate`	DateTime	R	`messageCreationDate`

sort

1	`Message.creationDate`	↓
2	`Message.id`	↑

limit	20
CPs	1.1, 2.2, 2.3, 3.2, 8.5
relevance	This is a navigational query looking for paths of length two, starting from a given Person, going to their friends and from them, moving to their published Posts and Comments. This query exercices both the optimizer and how data is stored. It tests the ability to create execution plans taking advantage of the orderings induced by some operators to avoid performing expensive sorts. This query requires selecting Posts and Comments based on their creation date, which might be correlated with their identifier and therefore, having intermediate results with interesting orders. Also, messages could be stored in an order correlated with their creation date to improve data access locality. Finally, as many of the attributes required in the projection are not needed for the execution of the query, it is expected that the query optimizer will move the projection to the end.