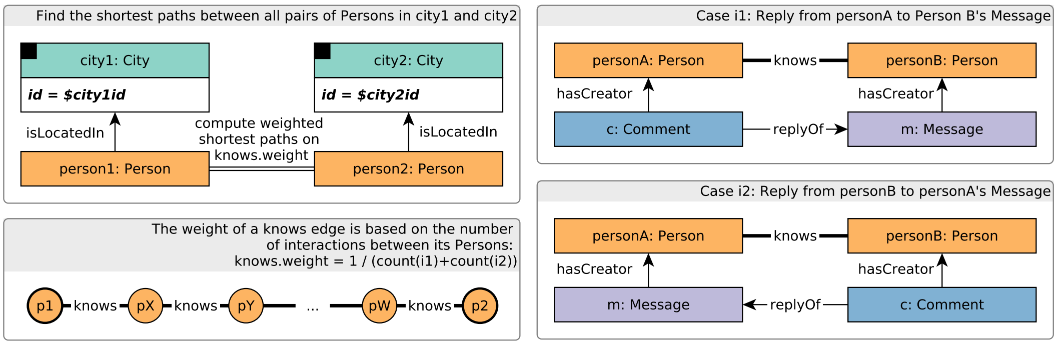
BI / read / 19

BI 1
BI 2
BI 3
BI 4
BI 5
BI 6
BI 7
BI 8
BI 9
BI 10
BI 11
BI 12
BI 13
BI 14
BI 15
BI 16
BI 17
BI 18
BI 19
BI 20

<table>
<tr><td>query</td><td>BI / read / 19</td></tr>
<tr><td>title</td><td>Interaction path between cities</td></tr>
<tr><td>pattern</td><td>
</td></tr>
<tr><td>desc.</td><td>Given two Cities city1, city2, find Persons person1, person2 living in these Cities (respectively) with the shortest <em>interaction path</em> between them. If there are multiple pairs of people with shortest paths having the same total weight, return all of them.
The shortest path is computed using a weight between two Persons defined as the reciprocal of the number of interactions (direct reply Comments to a Message by the other Person). Therefore, more interactions imply a smaller weight.
<em>Note:</em> Interactions are counted both ways, i.e. if Alice writes 2 reply Comments to Bob's Messages and Bob writes 3 reply Comments to Alice's Messages, their total number of interactions is 5.</td></tr>
<tr><td>params</td><td><table>
<tr><td>1</td><td>city1Id</td><td>ID</td><td>Small Cities within the same Country are selected</td></tr>
<tr><td>2</td><td>city2Id</td><td>ID</td><td></td></tr>
</table></td></tr>
<tr><td>result</td><td><table>
<tr><td>1</td><td>person1.id</td><td>ID</td><td>R</td><td></td></tr>
<tr><td>2</td><td>person2.id</td><td>ID</td><td>R</td><td></td></tr>
<tr><td>3</td><td>totalWeight</td><td>64-bit Float</td><td>C</td><td></td></tr>
</table></td></tr>
<tr><td>sort</td><td><table>
<tr><td>1</td><td>totalWeight</td><td>↓</td><td></td></tr>
<tr><td>2</td><td>person1.id</td><td>↑</td><td></td></tr>
<tr><td>3</td><td>person2.id</td><td>↑</td><td></td></tr>
</table></td></tr>
<tr><td>limit</td><td>20</td></tr>
<tr><td>CPs</td><td>3.3, 7.6, 7.7, 8.4, 8.6</td></tr>
<tr><td>relevance</td><td>Finding shortest paths between pairs of Persons in Cities can be implemented in theory with an <em>all-pairs shortest paths</em> algorithm. However, this needs to be executed on the whole Person-knows-Person graph (with edge weights derived from the number of interactions) so it is expected to be prohibitively expensive. A better approach is using multiple <em>single-source shortest path algorithms</em> (e.g. from the City with fewer inhibitants).</td></tr>
</table>