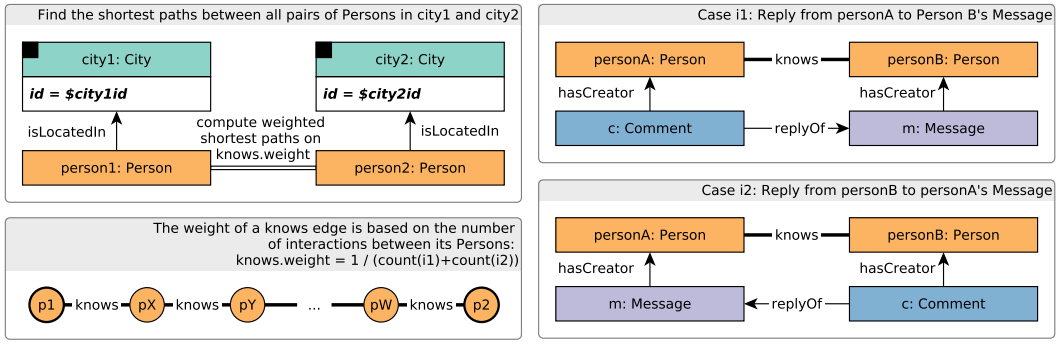
## BI / read / 19

| | |
|---|---|
| query | BI / read / 19 |
| title | Interaction path between cities |
| pattern |  |
| desc. | Given two Cities `city1`, `city2`, find Persons `person1`, `person2` living in these Cities (respectively) with the shortest *interaction path* between them. If there are multiple pairs of people with shortest paths having the same total weight, return all of them.<br>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.<br>*Note:* 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. |

**params**

| | | | |
|---|---|---|---|
| 1 | `city1Id` | ID | Small Cities within the same Country are selected |
| 2 | `city2Id` | ID | |

**result**

| | | | | |
|---|---|---|---|---|
| 1 | `person1.id` | ID | R | |
| 2 | `person2.id` | ID | R | |
| 3 | `totalWeight` | 64-bit Float | C | |

**sort**

| | | | |
|---|---|---|---|
| 1 | `totalWeight` | ↑ | |
| 2 | `person1.id` | ↑ | |
| 3 | `person2.id` | ↑ | |

| | |
|---|---|
| limit | 20 |
| CPs | 3.3, 7.6, 7.7, 8.4, 8.6 |
| relevance | Finding shortest paths between pairs of Persons in Cities can be implemented in theory with an *all-pairs shortest paths* 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 *single-source shortest path algorithms* (e.g. from the City with fewer inhibitants). |