Conceptual model

Aleksander Czeszejko-Sochacki

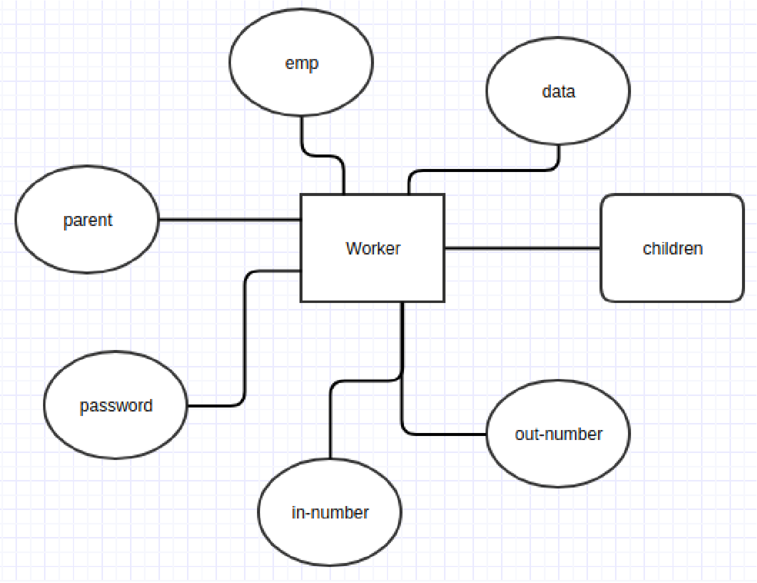
June 12, 2018

1 E-R description

Both root and app have privileges to all the fields in the database. In the diagram we can see the following fields:

1. Workers
   - emp - id of worker
   - data - some data of/about worker
   - parent - id of worker's direct supervisor
   - password - password related with worker
   - children - list of worker's direct subordinates
   - _in - in-order number of worker
   - _out - out-order number of worker

2 E-R Diagram



3 Basic idea

We consider the workers dependencies as a graph, more precisely, tree. Then we are able to store in and out numbers depending on the order of visiting during DFS algorithm. These numbers lead us to response in O(1) for the queries: ancestor, parent, child (having found the worker) and O(n) for the queries ancestors, descendants. We are obligated to update these numbers after each new/remove command. However, in normal mode, they are rare.

4 A brief description of API operations

The API should implement the following operations:

- root - creates a boss with no supervisor, need to add one worker to the Workers
- new - creates a worker with specified parameters, need to add one worker to the Workers and update parent's children
- remove - removes specified worker and all of his subordinates (both direct and indirect). Based on the _in and _out fields
- child - lists all of the worker's direct subordinates - using the children field
- parent - returns the worker's parent - using the parent field
- ancestors - lists all of the worker's supervisors (both direct and indirect) - based on the _in and _out fields
- descendants - lists all of the worker's subordinates (both direct and indirect) - based on the _in and _out fields
- ancestor - tells if the specified worker is direct or indirect supervisor of another specified worker - based on the _in and _out fields
- read - returns worker's data, using the data field
- update - updates worker's data in the the data field