

# Report Project 3 : XQuery

Rémy Detobel & Nathan Liccardo

April 12, 2018

## 1 Introduction

This document aims to detail each choices and hypothesis we made during our implementation. As a reminder, we were assigned to implement three different XQuery programs. Each of those query use a part of the BDLP database (DBLP-excerpt). The structure of this report will be divided into three parts (one per program).

## 2 First XQuery program

For the first program, we were assigned (for each author) to return the number of co-authors and the number of joint publications with each of them. To realise this, we start by iterate on each author as below :

```
• for $author in //author
  return <author>
      ....
  </author>
```

Which will create an `<author> ... </author>` bloc for each author. Inside each of these blocs, we will find following informations (in sequence).

### 2.1 Author name and Co-authors informations

The author name and the number of co-authors are respectively obtained using `data` and `count` functions (defined by XQuery). Below instructions are formal definitions of `name` and `coauthors` tags :

```
• <name>{data($author)}</name>
• <coauthors number="{count(//*[author=$author]/author)-1}"> ... </coauthors>
```

#### 2.1.1 Co-authors informations

For each co-author, we must obtain his name and the number of joint publications. Both informations will be contained inside `<coauthor> ... </coauthor>` tags (itself contained in the `coauthors` bloc). Each of the `coauthor` blocs are created by iterating on co-authors as below :

```
• for $coauthor in //*[author=$author]/author[not(.$author)]
  return <coauthor>
      ....
  </coauthor>
```

**Co-author name** As for the author name, co-author name is reached using the `data` function on the `coauthor` variable. This is realised as follow :

```
• <name>{data($coauthor)}</name>
```

**Joint publications** For each co-author, we must retrieve the number of joint publications. Once again, this is achieved using the `count` function available in XQuery. Here is the final instruction :

- `<nb_joint_pubs>{count(//*[author=$author]/author[.=$coauthor])}</nb_joint_pubs>`

### 3 Second XQuery program

For the second request, we were assigned to give (for each proceedings) its title and titles of articles appearing in it. Each `proceeding` blocs are created by the use of an iteration :

- ```
for $proceeding in //proceedings
  return <proceedings>
  ....
</proceedings>
```

For each of these blocs, we must retrieve the title (`proc_title`) and a list of articles (`title`). This is achieved using the value of the `key` attribute (inside `proceedings` tags) and the value contained inside `crossref` tags. Regarding to the title, we obtain the below instruction :

- `<proc_title>{data($proceeding/title)}</proc_title>`

Finally, we can create the list of titles using the `key` value :

- ```
for $inproceeding in //inproceedings[crossref=data($proceeding/@key)]
  return <title>{data($inproceeding/title)}</title>
```

### 4 Third XQuery program

For the last program, we were assigned to compute for each pair of authors  $x$  and  $y$  ( $x \neq y$ ) the distance between  $x$  and  $y$ . As we decided to use functions to simplify the request, we will firstly describe those functions before final instructions.