POLITECNICO
MILANO 1863

Project - Delivery II

Andrea Tocchetti
andrea.tocchetti@polimi.it

Project Delivery

Details about the second project delivery

- You have **two weeks** to deliver the second part of the project, starting **tomorrow**
  - MongoDB
- Continue to expand the document with the 1st delivery
- You will be asked to deliver the 2nd delivery on WeBeep by the **23rd of November** (actual deadline 24th of November at **1:00 A.M.**).
- After the 23rd of November, we will release the calendar for the final presentation

Data Structure

Full description of document structure and content

- Title, Abstract, Authors (with affiliations, email, bio)
- Metadata (keywords)
- Publication details (journal, volume, number, date, pages)
- Sections: title, text (by paragraph), subsections, figures (image URL and caption)
- Bibliography (set of refs.)

Project Delivery

The delivery should contain

- Description of the structure of a document
  - If you already described the attribute in the first delivery, feel free to copy-paste and update them (e.g., their type, description, etc.) and integrate the new ones
- Example(s) of document(s) and Description
- Data Upload and Transformation
  - Potential pre-processing, upload process, etc.

Project Delivery

- Queries (Cypher) !!! (Report)

mongoDB

!!! - **For each query**, report a **readable** picture/table with (a part) of the result, and a description of how the query works.

Try to be as **coherent** as possible with the 1st delivery regarding types, etc.

Neo4j Queries

The queries deliveries should include

- Minimum 5 (diverse) data creation/update commands
- Minimum 10 queries (15 in total, with the update/creation)
- Minimum complexity of queries:
  - Multiple Filtering conditions
  - Multiple Filtering conditions, aggregation
  - Multiple Filtering conditions (also on internal subdocs)
  - Multiple Filtering through unwind
  - Multiple Filtering through refs
- Deliver a **mix** of them.
- Check complexity / performance time

FAQ - Frequently Asked Questions

How many query should I make for each type?

- My advice would be to evenly split the 10 query between the 5 query complexities

Can I improve the previous deliveries?

- Yes, you can. Keep in mind that **only the original delivery will be evaluated**! In the end, improving the final document is useful to get the final bonus (Max 3 points).
- If you apply any updates, **mark** them in a different color (e.g., red) so that I can immediately notice them when comparing the different versions of the document.
- We will not let you know the mistakes you made in the first delivery. Consider this opportunity as a way to get extra time to work on it.
- Highlight and focus on important changes, minor ones (e.g., grammar) won't be considered.

MongoDB - Joins

In MongoDB, we can perform join between different collections (or the same collection) by using the **$lookup** operator.

from _id to for_id

To perform joins (as you may already know), it is highly preferred to use a unique identifier, in MongoDB, the **_id** represents such an element.

Therefore, it is necessary to set the "**Foreign Keys**" in documents. To do so, it is necessary to report the exact same **_id** of a document as a field of another one. This approach is called **Manual Reference** which is highly preferred to **DBRefs** (another more complex reference type).

MongoDB - Joins

A join B on A._id = B.foreign_id

The **$lookup** operator is applied like any other operator within the query pipeline, in particular

- **from** – The name of the **local** collection.
- **localField** – The name of the field in the **local** collection that will be used to perform the join.
- **foreignField** – The name of the field in the **target** collection that will be used to perform the join.
- **as** – The name of the field that will be created by the **$lookup** operator.

The operator will return a **collection** (i.e., an **array**) containing all the elements that are matched during the lookup.

It is not required to manually manage joins between a **local** array and a different target documents as it is already managed by the operation itself.

MongoDB - Joins

```
db.collection1.aggregate([{
        $lookup: {
                from: collection1
                localField: _id
                foreignField: foreign_id
                as: join_result
        }
}])
```

The **$lookup** operator allows for an even more complex set of operations. Some of them may even simplify the query after the lookup! Have a look at them here.