POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Systems and Methods for Big and Unstructured Data Project

Author(s): **Simona Malegori**
Nicole Perrotta
Michele Simeone
Alberto Pirillo
Simone Tognocchi

Group Number: **38**

Academic Year: 2022-2023

Contents

1 | First Delivery

1.1. Problem description

This project aims at building an Information System that manages a data set containing different type of scientific articles that can be used for: clustering with network and side information, studying influence in the citation network, finding the most influential papers and topics, modeling analysis.

The project is divided in the following steps.
At first it was made an ER Diagram that generalizes all the information gathered from different already existing data sets, then the most complete data set was chosen.
Afterwards the data set was pre-processed, transforming it from a JSON to a CSV format and then it was reduced in size.
After that it was uploaded on Neo4j and all the nodes, the relationships and the properties were edited to build the Graph Diagram.
At the end of the project 10 queries and 6 creation/update commands were initiated with a different level of complexity that was checked within the performance time.

1.2. Hypothesis

In order to model the database we made some assumptions:

- authors can have one or more papers associated
- authors can have zero, one or more affiliated organizations, assuming that there can be authors that didn't provide their organization;
- papers have at least one author;
- papers have at least one field of study;
- not all papers have keywords associated, assuming that they may not have been provided;

- papers can reference and be referenced by other papers;
- each paper has a venue, that is the place where it has been published/presented;
- a venue can host more than one paper;
- venues are of 4 types: Journal, Conference, Book and Patent;
- the volume n of a paper is the n-*th* published collection;
- the issue m of a paper is the m-*th* part of the volume in which it is published.

1.3. Data

1.3.1. ER Diagram

The ER Diagram of the chosen model is characterized by the following entities with the respective attributes:

- **Paper** is a scientific article that is associated with the following attributes: *id*, *title*, *date* that corresponds to the publication date, *doi* that is the Digital Object Identifier, *volume* that corresponds to the n-th published collection, *issue* that corresponds to the m-th part of the volume, *language*, *issn* that is an identification code associated with the title of the publication, *isbn* that is a code that identifies printed or digital papers and it is used as inventory-tracking device, *n_citation* that is the number of citations, *page_start* and *page_end* that are the starting and the ending point of the collection from which the paper was extracted, *pdf_url* that is the source from which to recover the paper, *abstract* that is the summary of the paper, *publisher* and *external_url* that corresponds to the sitography of the paper;
- **Author** with the attributes: *id*, *name*, *surname*, *email*, *orcid* that is a unique and persistent identification number and *organization*;
- **Keyword** that represents a word that allows to define immediately the topic within the paper, its attributes are: *id* and *name*;
- **FoS** that corresponds to the field of studies with the attributes: *id*, *name*, *w* that is the weight of the fields of study;
- **Venue** that is the collection from which the paper was extracted, with the attributes: *id* and *name*. The venue of the paper can be of different types, in fact there is a total and exclusive generalization of the entity Venue that can be a:

  - **Journal** that has also the attribute *addressee* that is the type of audience of the paper;
  - **Book** that has also the attributes *category* and *edition*;
  - **Conference** that has also the attributes *type* that can be physical and online, and *location* that has cardinality one only if the type is physical and it represents the place in which the conference takes place;
  - **Patent** that has also the attributes *type* and *expiration*.

In the ER Diagram there are the following relationships with the respective cardinalities:

- **Writing** between the entities Paper and Author. The relationship means that a Paper can be written by at least 1 to a maximum of N authors and that an Author can write at least 1 to a maximum of N papers.
- **Containing** between the entities Paper and Keyword. The relationship means that a Paper can contain from 0 to N keywords and that a Keyword is contained into at least 1 to a maximum of N papers.
- **Dealing** between the entities Paper and FoS. The relationship means that a Paper can deal with at least 1 to a maximum of N field of studies and that a FoS can be dealt from 0 to N papers.
- **In** between the entities Paper and Venue. The relationship means that a Paper is extracted from exactly 1 venue and that a venue can be the collection in which at least one paper is contained.
- **Referencing** that is a relationship on the same entity of the Paper, in fact it contains the roles referencer and referenced. The relationship means that a Paper can or not reference other Papers and can be referenced or not from other Papers.

After all, in the ER Diagram there is an external constraint on the attributes of the entity Conference. In fact, the attribute *location* must have a cardinality of (1,1) if the type is *physical*.

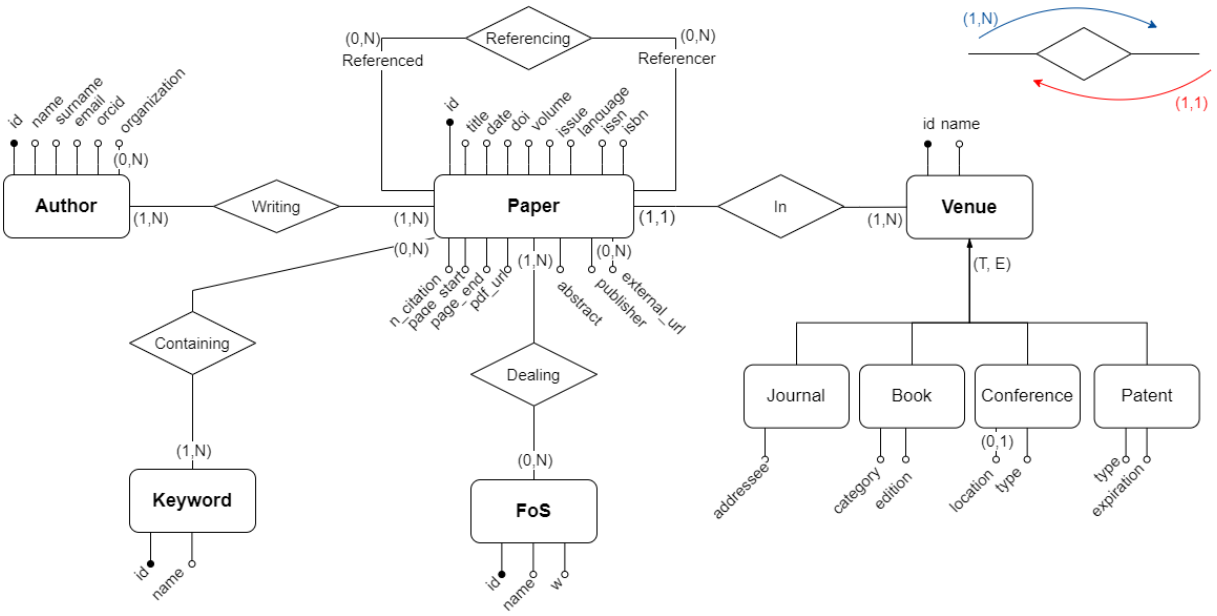


Figure 1.1: ER Diagram

errata corrige:

- added cardinality also on the left side of the Referencing relationship
- changed cardinality of the left side of the Writing relationship from (0,N) to (1,N)

1.3.2. Data set description

The data set that was chosen for the goal of the project contains 8335 papers, 730 venues, 27576 authors and 25005 field of studies. It is a reduced version of the more complex model on which the ER Diagram is based. In fact, it doesn't contain the entities Keyword and the sub-entities Journal, Book, Conference and Patent that, instead, are transformed into an attribute of the entity Paper that is called doc_type. All the respective attributes of these entities are eliminated. Moreover, the attribute *date* of the Paper, that is generalized in the ER Diagram, becomes the year. Furthermore, in the reduced version of the data set, the relationship Referencing becomes a list of string that are the references of the Paper. Lastly, the chosen data set contains just the more significant attributes. Summarizing, the chosen data set is composed by the following entities and attributes:

- **Paper** with the attributes: id (integer type), title (string type), year (integer type), n_citation (integer type), page_start (integer type), page_end (integer type), doc_type (string type), publisher (string type), volume (integer type), issue (integer type), doi (string type), references (list of string type), abstract (string type);

- **Author** with the attributes: paper_id (integer type), author_name (string type), author_id (integer type), author_org (string type);
- **FoS** with the attributes: paper_id (integer type), fos_name (string type), fos_weight (float type);
- **Venue** with the attributes: paper_id (integer type), venue_name (string type), venue_id (integer type).

1.3.3. Data Pre-Processing

The original data set can be downloaded *here*. Given that such data set was unnecessarily large for our purpose, we decided to reduce its size. We could have just cut it at a certain point, however we decided that it was better to work with consistent data, therefore we decided to carefully perform sub-sampling in an intelligent way. Such pre-processing was performed using multiple Python scripts and the Pandas library.

Here we provide a description of all the scripts used, together with the procedure to obtain the final data set starting from the initial one. All of these scripts can be found in the *scripts* folder of the *neo4j* section of the *GitHub repository* of the project.

The notebook **dataset_exploration.ipynb** contains a short description of every operation for explanatory reasons. This notebook processes only a small chunk of the data set. The same operations are performed on the whole data set in the script **dataset_preprocessing.py.**

Here is a summary of the operations performed:

- Removal of samples with Null and NaN values
- Removal of samples with an empty string in a field

The operations above are required to work with consistent data. Notice that we can afford to simply drop the samples that do not respect such conditions since we dispose of a very large data set.

The following operations are not required but were performed to reduce even further the size of the data set, with the objective of keeping only the "most important" samples.

We kept the samples:

- With a number of citations greater than a threshold
- With a reference count greater than a threshold

We converted the *indexed_abstract* field from an inverted index to a string of text, to make it easier to query once inserted into the database. The field was also renamed to *abstract*.

We also processed the data set in order to remove some special characters not supported by the import function of the database. An example of such characters is "\". This operation is performed in the script **remove_special_characters.py**.

Given that many samples were removed from the data set, we also had to fix up the *references* field to keep only the valid ones. We consider a reference valid when it points to a sample of the data set that was kept. Otherwise, we say that such reference is invalid and we remove it from the data set. This operation is performed in the script **deprecated_references.py**.

Lastly, when importing the data into the database, we realized that it was more practical to split the data set into multiple data sets to speed up the process and to produce cleaner code. This functionality was added in the **dataset_preprocessing.py** script. We split the data set following the structure of ER diagram, ending up with one separate data set for each entity present in the original data set. Thus, we ended up with 4 data sets: **Paper**, **Author**, **Venue** and **FoS**.

To obtain the final data set starting from the downloaded one, run the scripts in this order:

1. **dataset_preprocessing.py**
2. **deprecated_references.py**
3. **remove_special_characters.py**

The input of the first script is the initial data set. Only the paper data set requires to be processed by the second script, then only the paper and the venue data sets require to be processed by the third script. At the end, you will obtain 4 data sets which are identical to the ones that we imported into the database.

1.4. Neo4j

1.4.1. Data Upload

To import the data into Neo4j, there is one last precaution needed: it is necessary to process the Paper data set to make the *references* field compatible with the import function of the database. To perform such action the script **preprocessing.py** can be used. The script is located in the *neo4j* folder of the *GitHub repository*. Now, the data sets can be correctly imported into the Neo4J database. In order to do that, you need to put the four data sets files in the program's import folder.

Thanks to the pre-processing that was performed, the four data sets are saved in the simple CSV format and it is possible to use the LOAD CSV command to easily load all the entities and relationships.

1. Clear the Database:

```
MATCH (x) DETACH DELETE x;
```

2. Load the Papers:

```
LOAD CSV WITH HEADERS FROM "file:///paper_dataset2.csv" AS csvLine
CREATE (p:Paper {id: toInteger(csvLine.id), title: csvLine.title,
   year: toInteger(csvLine.year), doi: csvLine.doi, volume: csvLine
   .volume, issue: csvLine.issue, abstract:csvLine.abstract,
   n_citation:toInteger(csvLine.n_citation), page_start:toInteger(
   csvLine.page_start), page_end:toInteger(csvLine.page_end),
   publisher:csvLine.publisher, doc_type:csvLine.doc_type})
```

3. Load the Authors:

```
LOAD CSV WITH HEADERS FROM "file:///author_dataset.csv" AS csvLine
CREATE (a:Author {id: toInteger(csvLine.author_id), name: csvLine.
   author_name, organization: csvLine.author_org})
```

4. Load the FoS:

```
LOAD CSV WITH HEADERS FROM "file:///fos_dataset_0.csv" AS csvLine
CREATE (f:Fos {weight: toFloat(csvLine.fos_weight), name: csvLine.
   fos_name})
```

5. Load the Venue (*errata corrige*: CREATE instead of LOAD at line 2):

```
LOAD CSV WITH HEADERS FROM "file:///venue_dataset.csv" AS csvLine
CREATE  (v:Venue {id: toInteger(csvLine.venue_id), name: csvLine.
   venue_name})
```

6. Create the `Writing` relationship between papers and authors:

```
LOAD CSV WITH HEADERS FROM "file:///author_dataset.csv" AS csvLine
MATCH (p:Paper),(a:Author)
WHERE (toInteger(csvLine.paper_id)=p.id) and (toInteger(csvLine.
    author_id) = a.id)
CREATE (p)-[w:writing]->(a)
```

7. Create the `Referring` relationship between papers and papers:

```
LOAD CSV WITH HEADERS FROM "file:///paper_dataset2.csv" AS csvLine
UNWIND split(csvLine.references,':') as ref
MATCH (p1:Paper),(p2:Paper)
WHERE (toInteger(csvLine.id)=p1.id) and (toInteger(ref)=p2.id)
CREATE (p1)-[r:referencing]->(p2)
```

8. Create the `in` relationship between papers and venues:

```
LOAD CSV WITH HEADERS FROM "file:///venue_dataset.csv" AS csvLine
MATCH (p:Paper),(v:Venue)
WHERE (toInteger(csvLine.paper_id)=p.id) and (toInteger(csvLine.
    venue_id) = v.id)
CREATE (p)-[i:in_]->(v)
```

9. Create the `dealing` relationship between papers and FoS:

```
LOAD CSV WITH HEADERS FROM "file:///fos_dataset_0.csv" AS csvLine
MATCH (p:Paper),(f:Fos)
WHERE (toInteger(csvLine.paper_id)=p.id) AND (csvLine.fos_name = f.
    name) AND (toFloat(csvLine.fos_weight) = f.weight)
CREATE (p)-[d:dealing]->(f)
```

1.4.2. Graph Diagram

Entities:

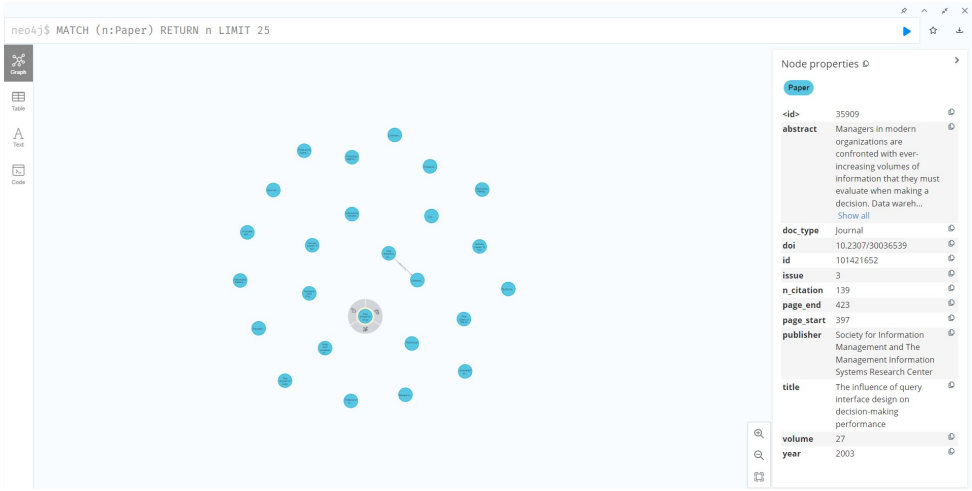


Figure 1.2: Paper

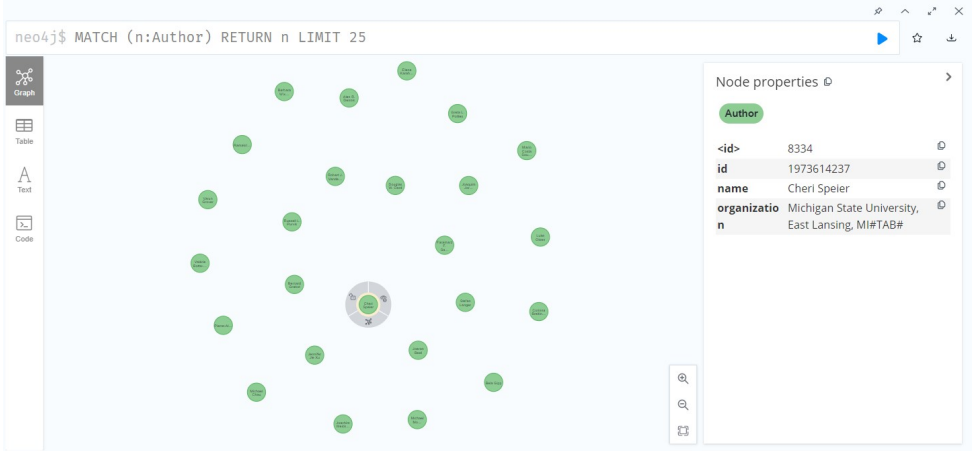


Figure 1.3: Author

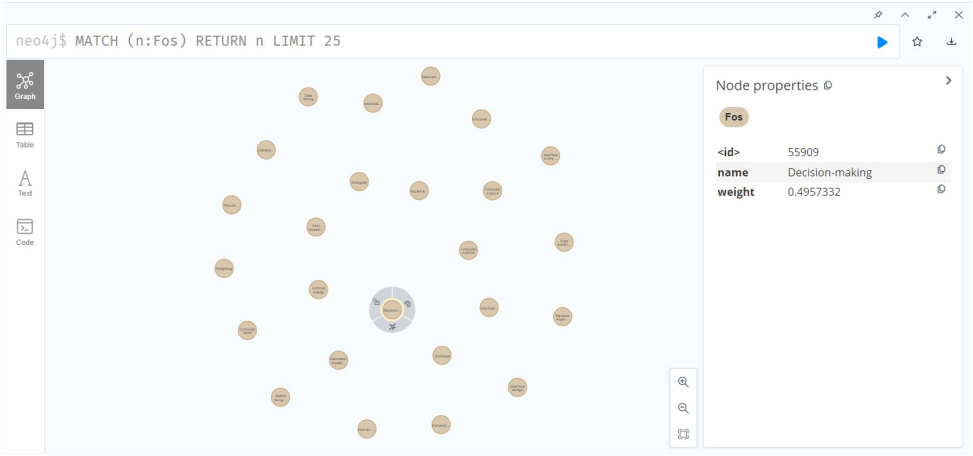


Figure 1.4: FoS

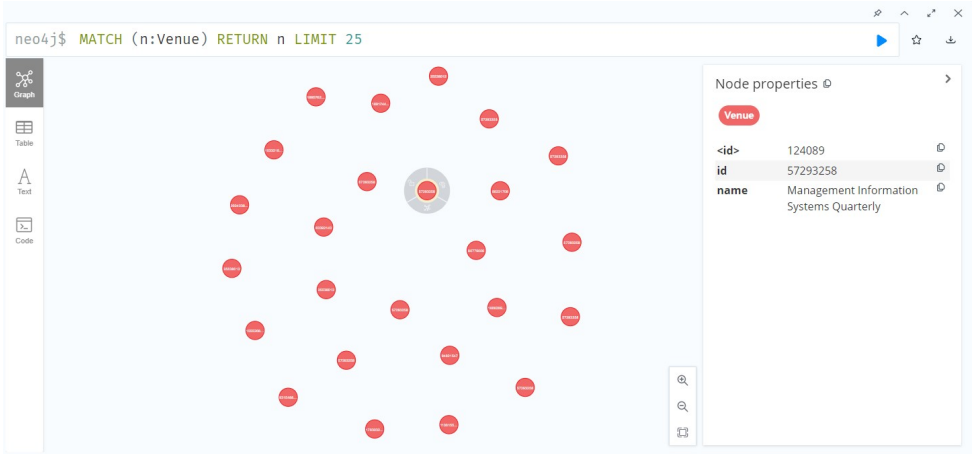


Figure 1.5: Venue

Relationships:

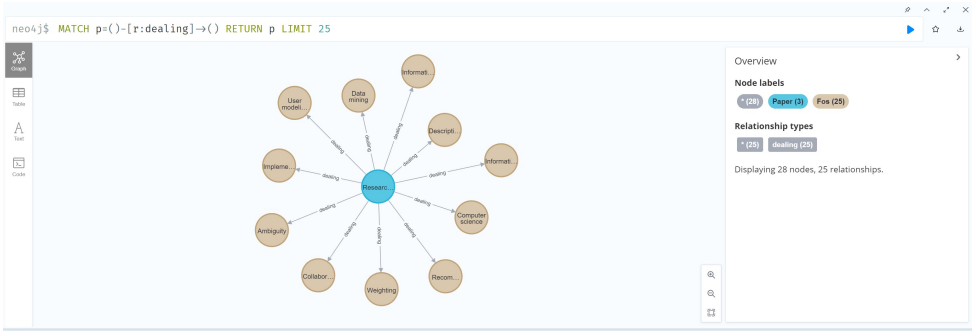


Figure 1.6: Paper-Dealing->FoS

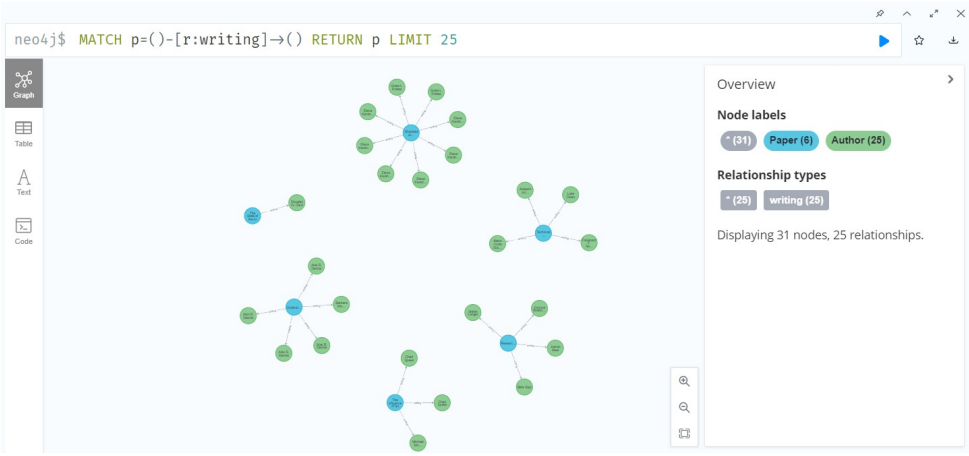


Figure 1.7: Paper-Writing->Author

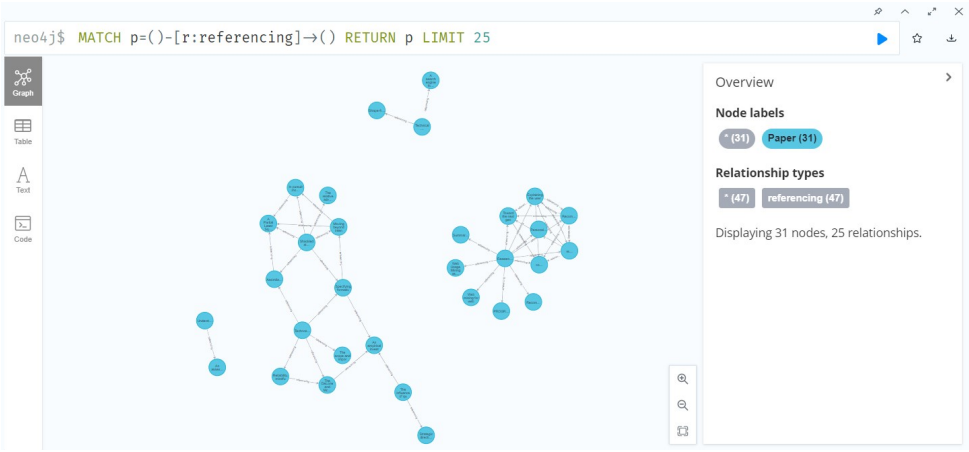


Figure 1.8: Paper-Referencing->Paper

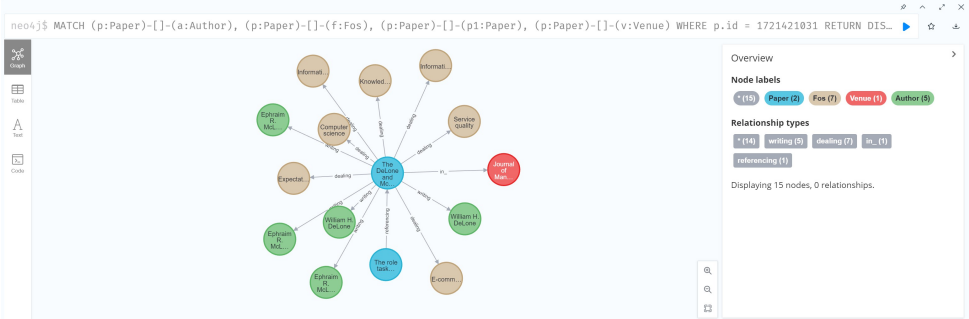
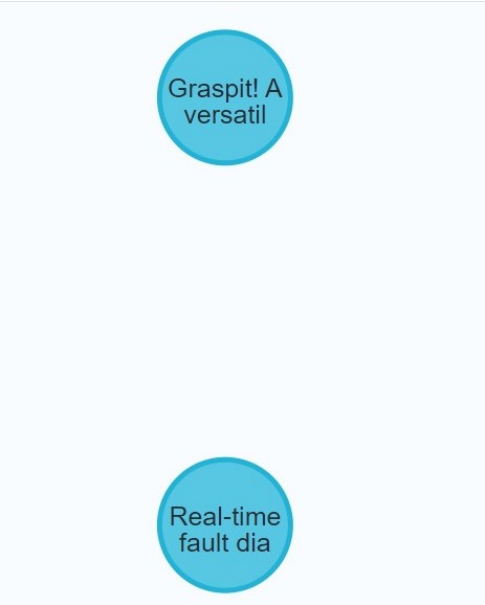


Figure 1.9: All the relationships

1.4.3. Queries

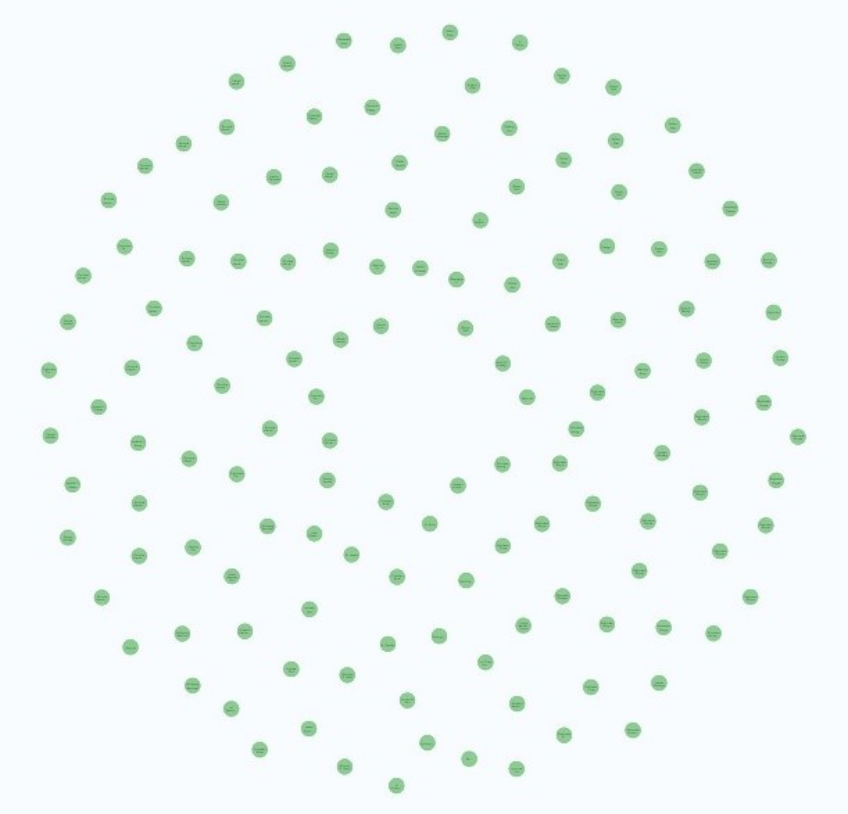
1. **Find papers of a determined venue, written after a certain date (Execution time 454ms):**

```
MATCH (p: Paper)-[i:in_]->(v: Venue), (p: Paper)-[d:dealing]->(f:
    Fos)
WHERE (p.year > 2000) AND (v.name="IEEE Robotics & Automation
    Magazine")
RETURN p
```

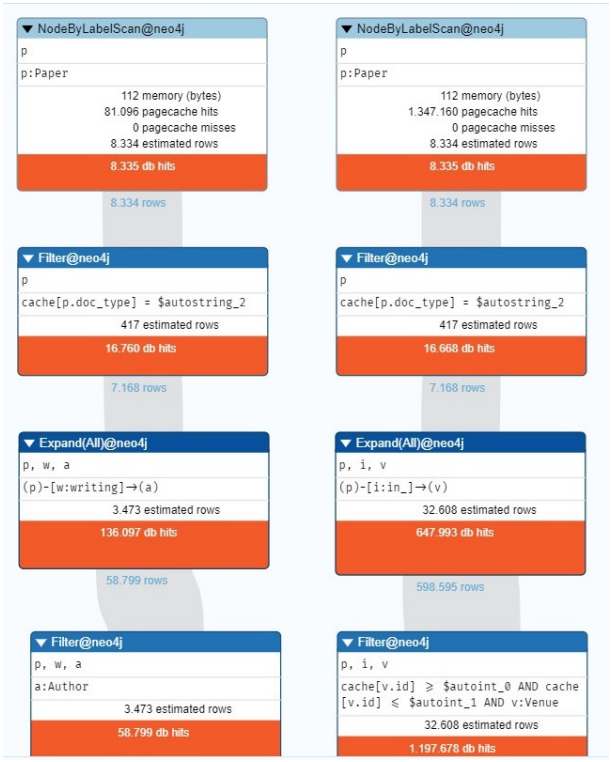


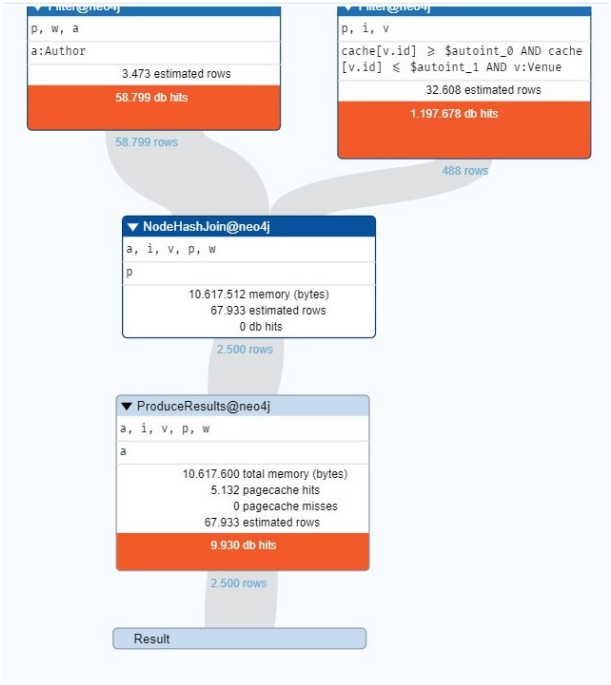
2. **Find authors that wrote papers in a set of venues, with a determined type (Execution time 2ms):**

```
MATCH (p: Paper) -[w:writing]->(a:Author), (p: Paper) -[i: in_]->
    (v:Venue)
WHERE v.id >= 140000000 and v.id <= 140900000 AND p.doc_type='
    Journal'
RETURN a
```



Below we can see the output of the profile statement, that is used to track the query and the numbers of rows of each operation. It starts by scanning all the nodes with the papers, then it expands all the nodes with the 'writing' relationship on the left of the picture and the 'in' relationship on the right. Finally, it applies the filters on the venue id and on the paper's doc type and returns the results.

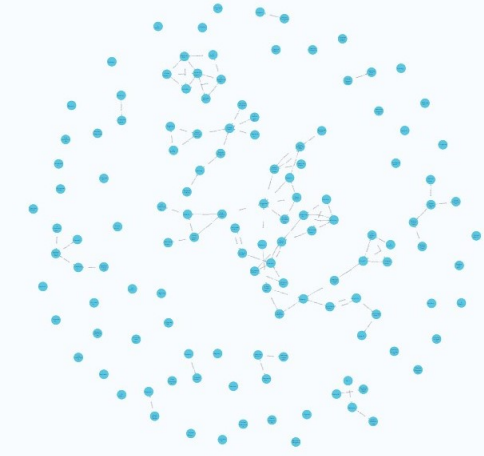




3. **Find papers of a determined main argument, written after a certain date (Execution time 30ms)**:

```
MATCH (p: Paper)-[r:referencing]->(p2: Paper), (p2: Paper)-[d:
    dealing]->(f: Fos)
```

```
WHERE (f.weight >= 0.45) AND (p2.year >2010) AND (f.name="
    Artificial intelligence")
RETURN p
```



4. **Count the authors that have written a famous paper of a determined argument (Execution time 38ms):**

```
MATCH (p: Paper)-[w:writing]->(a:Author), (p: Paper)-[d: dealing]->(f:
    Fos)
WHERE f.name = "Machine learning" AND p.n_citation>5000
RETURN COUNT(a.id) AS num_aut
```

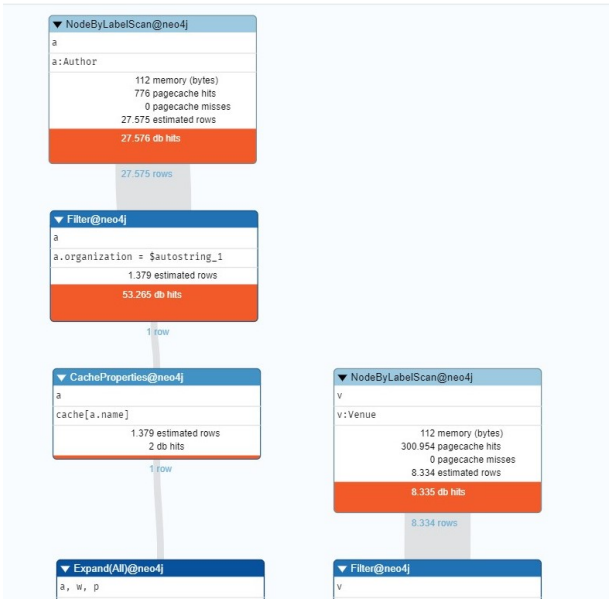
	num_aut
1	2120

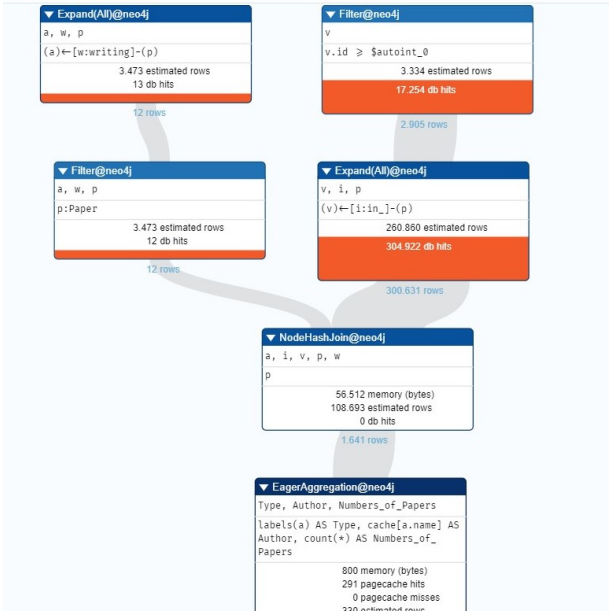
5. **Count the papers divided per author written in a set of venue by a determined authors' organization (Execution time 103ms):**

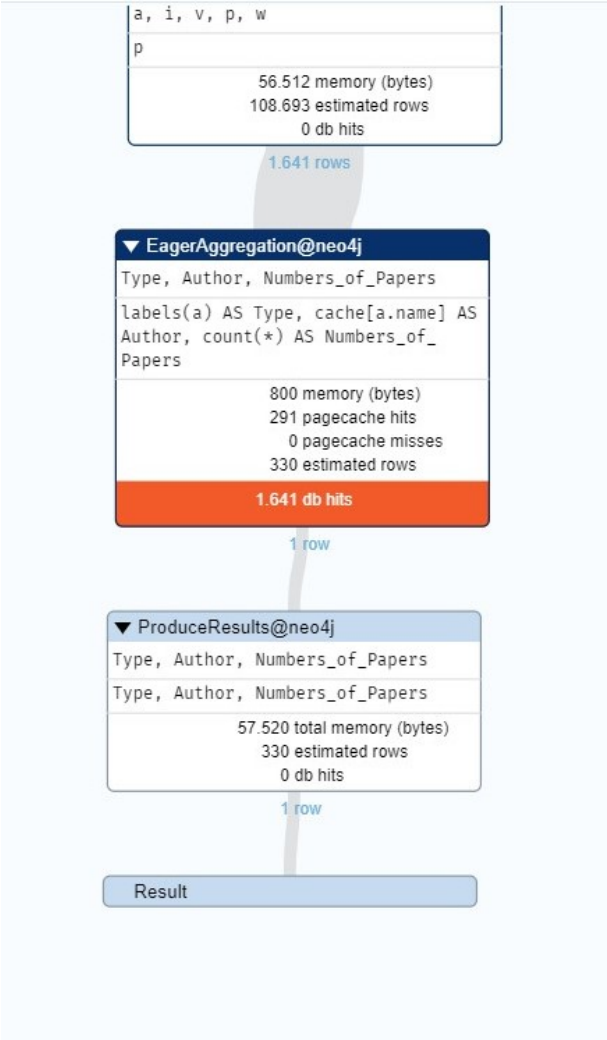
```
MATCH (p: Paper) -[w:writing]->(a:Author), (p: Paper) -[i: in_]->(v
    : Venue)
WHERE v.id >= 160000000 AND a.organization='Microsoft Research, One
    Microsoft Way, Redmond, WA 98052, USA. hshum@microsoft.com#TAB#
    '
RETURN labels(a) AS Type, a.name AS Author, COUNT(*) AS
    Numbers_of_Papers
```

	Type	Author	Numbers_of_Papers
1	["Author"]	"Heung-Yeung Shum"	1641

The profile statement scans all the authors, then it expands the 'writing' relationship and applies the organization's filter. At the same time it scans all the venues and filters their id. At the end the aggregation is applied and all the resulting values are returned.







6. **Calculate the weight keywords' average of papers that have a determined publisher and a reference to a famous paper (Execution time 27ms):**

```
MATCH (p: Paper) -[d: dealing]->(f: Fos), (p: Paper) -[r:
    referencing]->(p2: Paper)
WHERE p2.n_citation>2000 AND p.publisher='Society for Information
    Management and The Management Information Systems Research
    Center'
RETURN p.title, avg(f.weight)
```

	p.title	avg(f.weight)
1	"Shackled to the status quo: the inhibiting effects of incumbent system habit, switching costs, and inertia on new system acceptance"	0.449569698283237
2	"Technostress: technological antecedents and implications"	0.44386502606134975
3	"Business intelligence in blogs: understanding consumer interactions and communities"	0.41906192304436024
4	"Web and wireless site usability: understanding differences and modeling use"	0.42455533500106846
5	"Competing perspectives on the link between strategic information technology alignment and organizational agility: insights from a mediation model"	0.4178663593780648
6	"Reliability, mindfulness, and information systems"	0.42862394017677813

7. **Count the numbers of bilateral reference between two papers that have the same venue and a large number of citation (Execution time 119ms):**

```
MATCH (p: Paper)-[r:referencing]->(p2: Paper), (p2: Paper)-[r2:
    referencing]->(p: Paper), (p: Paper)-[w:writing]->(a: Author),(p
    : Paper)-[i:in_]->(v: Venue), (p2: Paper)-[i2:in_]->(v2: Venue)
WHERE p2.n_citation>500 AND p.n_citation>500 AND v.name=v2.name AND
     a.organization="Royal Institute of Technology"
RETURN type(r) AS Relation, COUNT(*)/2 AS
    Num_of_bilateral_referencingsame_venue, a.organization AS
    organization
LIMIT 1
```

	Relation	Num_of_bilateral_referencing_same_venue	organization
1	"referencing"	28900	"Royal Institute of Technology"

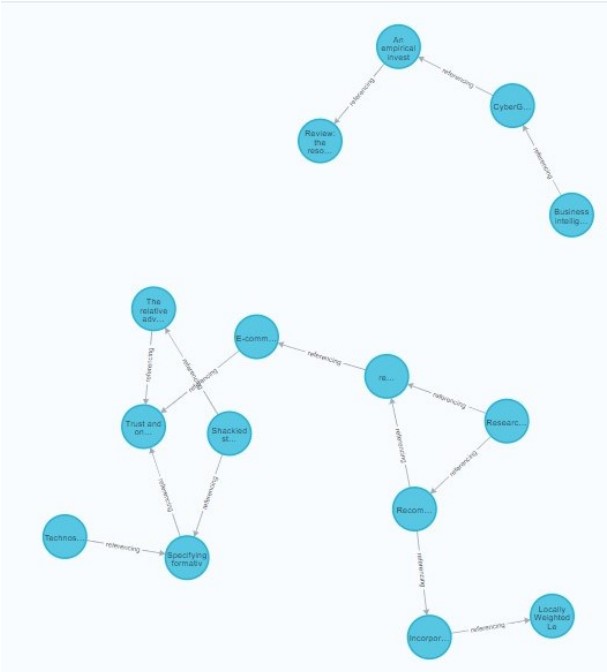
8. **Count the total citation of a paper that have references to two papers that deal of different field of study (Execution time 1849ms):**

```
MATCH (p: Paper)-[r: referencing]->(p2: Paper), (p: Paper)-[r2:
    referencing]->(p3: Paper), (p2: Paper)-[d: dealing]->(f: Fos), (
    p3: Paper)-[d2: dealing]->(f2: Fos)
WHERE p.page_end-p.page_start>10 AND p2 <> p3 AND f.name='
    Artificial Intelligence' AND f.weight>0.0 and f2.name = 'Machine
     learning' AND f2.weight>0.0
RETURN p.title AS Title, SUM(p.n_citation) AS Sum_n_citation
LIMIT 5
```

	Title	Sum_n_citation
1	"The SUN Attribute Database: Beyond Categories for Deeper Scene Understanding"	890400
2	"Composition in Distributional Models of Semantics"	4053440
3	"Wikipedia-based semantic interpretation for natural language processing"	2170880
4	"Knowledge derived from wikipedia for computing semantic relatedness"	737760
5	"A survey of paraphrasing and textual entailment methods"	4324800

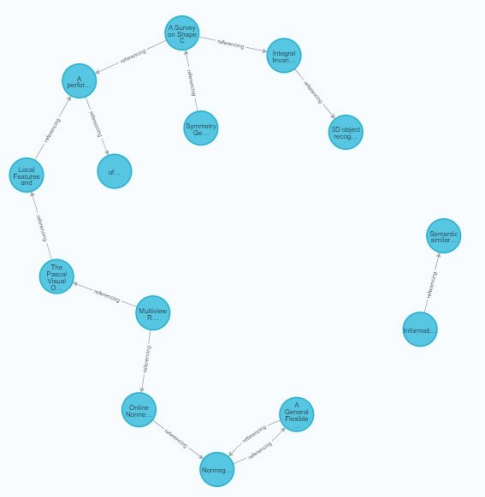
9. **Find the shortest path between two different paper that have a reference in common where the first is less famous than the second one (Execution time 61ms):**

```
MATCH s = shortestPath(
(p: Paper)-[r:referencing*]->(p2: Paper)
)
WHERE p2.n_citation>10*p.n_citation AND p <> p2
RETURN s
LIMIT 5
```

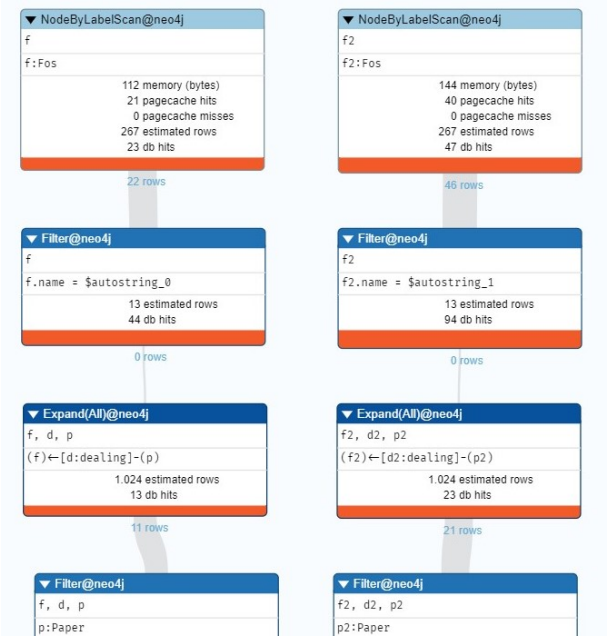


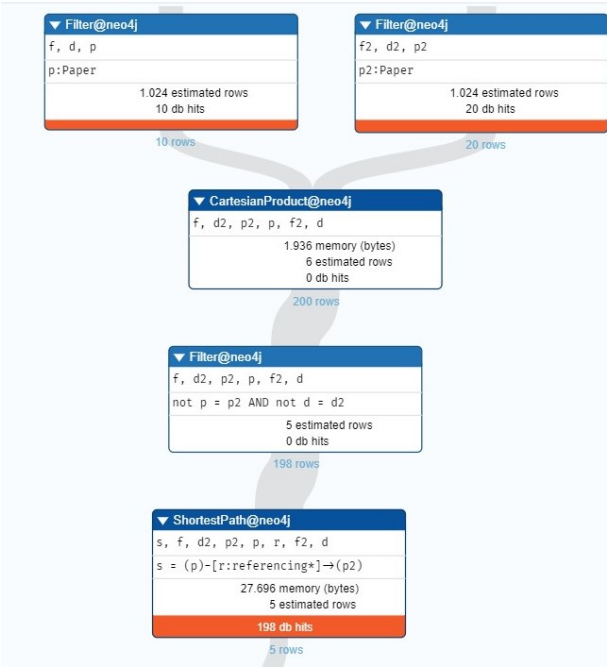
10. **Find the shortest path between two different paper that have a reference in common and different FoS (Execution time 42ms)**:

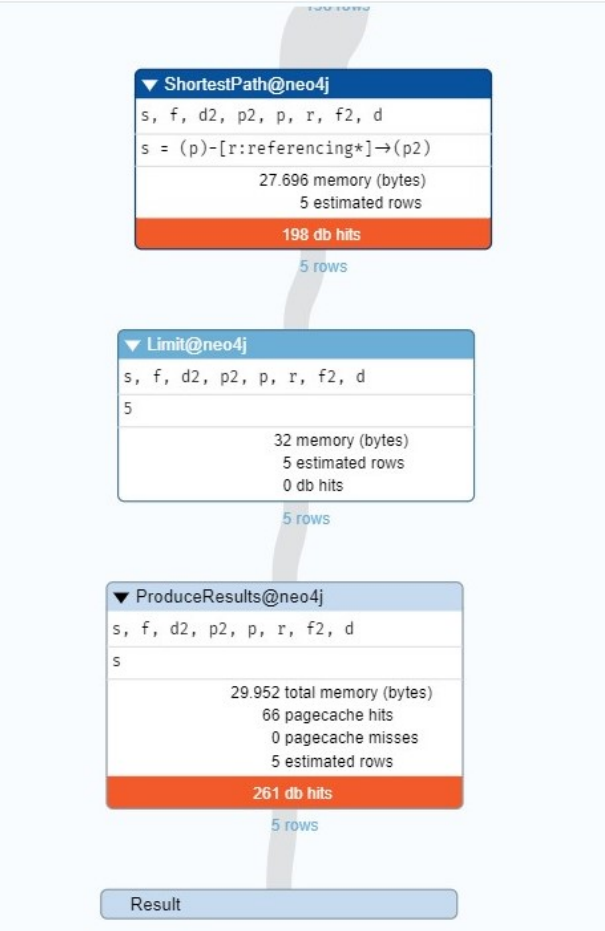
```
MATCH (p: Paper)-[d: dealing]->(f:Fos), (p2: Paper)-[d2:dealing]
   ->(f2:Fos), s = shortestPath( (p: Paper)-[r:referencing*]->(p2:
   Paper) )
WHERE f.name = "Artificial intelligence" AND f2.name = "Machine
   learning" AND p <> p2
RETURN s
LIMIT 5
```



In the following page we can see that the profile statement scans the FoS, applies the filters to them and expands the 'dealing' relationship. Then it checks that paper1 and paper2 are different and in the end it calls the function 'shortestPath' and returns the results.


NodeByLabelScan@neo4j
f
f:Fos
112 memory (bytes)
21 pagecache hits
0 pagecache misses
267 estimated rows
23 db hits
22 rows
NodeByLabelScan@neo4j
f2
f2:Fos
144 memory (bytes)
40 pagecache hits
0 pagecache misses
267 estimated rows
47 db hits
46 rows
Filter@neo4j
f
f.name = $autostring_0
13 estimated rows
44 db hits
0 rows
Filter@neo4j
f2
f2.name = $autostring_1
13 estimated rows
94 db hits
0 rows
Expand(All)@neo4j
f, d, p
(f)←[d:dealing]-(p)
1.024 estimated rows
13 db hits
11 rows
Expand(All)@neo4j
f2, d2, p2
(f2)←[d2:dealing]-(p2)
1.024 estimated rows
23 db hits
21 rows
Filter@neo4j
f, d, p
p:Paper
Filter@neo4j
f2, d2, p2
p2:Paper


Filter@neo4j
f, d, p
p:Paper
1.024 estimated rows
10 db hits
10 rows
Filter@neo4j
f2, d2, p2
p2:Paper
1.024 estimated rows
20 db hits
20 rows
CartesianProduct@neo4j
f, d2, p2, p, f2, d
1.936 memory (bytes)
6 estimated rows
0 db hits
200 rows
Filter@neo4j
f, d2, p2, p, f2, d
not p = p2 AND not d = d2
5 estimated rows
0 db hits
198 rows
ShortestPath@neo4j
s, f, d2, p2, p, r, f2, d
s = (p)-[r:referencing*]→(p2)
27.696 memory (bytes)
5 estimated rows
198 db hits
5 rows



1.4.4. Creation/Update Commands

1. **Insertion of a new author in the database**:

```
CREATE (a:Author {id: 3000000000, name:"Mario Rossi", organization:
    "Politecnico di Milano"})
```

2. **Insertion of a new field of study in the database**:

```
CREATE (f:Fos {weight:0.57640203, name:"Ambient intelligence" })
```

3. **Insertion of a new venue in the database**:

```
CREATE (v:Venue { id:3000000001, name:"Journal of Cloud Computing"
    })
```

4. **Insertion of a new paper in the database, paper's relationships with authors, venue, fields of study and other papers are created**:

```
MATCH (a:Author) WHERE a.id=3000000000
MATCH (f:Fos) WHERE f.name="Cloud computing" AND f.weight=0.6410412
```

```
MATCH (v:Venue) WHERE v.id=3000000001
MATCH (ref:Paper) WHERE ref.doi="10.1109/JPROC.2015.2483592"
MATCH (cit:Paper) WHERE cit.doi="10.1007/s10845-008-0158-5"
CREATE (p:Paper { id: 3000000000, title: "Application of
   deterministic, stochastic, and hybrid methods for cloud provider
    selection", year: 2022, doi: "10.1186/s13677-021-00275-1",
   volume: "11" , issue: "1" , abstract: "Cloud Computing
   popularization inspired ... requests.", n_citation: 5,
   page_start: 5, page_end: 10, publisher: "SpringerOpen", doc_type
   : "Journal" })
CREATE (p)-[w:writing]->(a), (p)-[r:referencing]->(ref), (cit)-[r:
   referencing]->(p), (p)-[i:in_]->(v), (p)-[d:dealing]->(f)
```

5. **Update of the organization of an author**:

```
MATCH (a:Author {name: 'Stefano Ceri'})
SET a.organization = "Politecnico di Milano"
```

6. **Update of the number of citations of a paper**:

```
MATCH (p:Paper {title: "Markov localization for mobile robots in
   dynamic environments"})
SET p.n_citation = 728
```

2 | Second Delivery

2.1. Introduction

This project aims at building an Information System that manages a data set containing different type of scientific articles that can be used for: clustering with network and side information, studying influence in the citation network, finding the most influential papers and topics, modeling analysis.

For the same problem described in Chapter 1 (First Delivery), we modeled a different system following the subsequent steps.
At first the data set was pre-processed to add the fields that were missing in the original data set. Such fields were taken from different Kaggle data sets. Then it was reduced in size and after that it was uploaded on MongoDB.
At the end of the project 11 queries and 6 creation/update commands were initiated with a different level of complexity that was checked within the performance time.

2.2. Data

This is the structure of a paper inside the database, it contains different sub-documents: authors, fos (metadata), venue and sections.

```
{
  "id": ...,
  "title": "...",
  "abstract": "...",
  "authors":[{
    "name": "...",
    "org": "...",
    "email": "...",
    "bio": "..."
    }],
```

```
  "fos":[{
    "name": "...",
    "weight": ...
    }],
  "year": ...,
  "page_start": ...,
  "page_end": ...,
  "doc_type": "...",
  "publisher": "...",
  "volume": ...,
  "issue": ...,
  "doi": "...",
  "n_citation": ...,
  "venue":{
    "raw": "...",
    "id": "...",
    },
  "references": [ ... ],
  "sections":[
      {
      "id": ...,
      "title": "...",
      "text": "...",
      "subsections": [{
          "id":...,
          "title": "...",
          "text": "..."
          }],
      "figures": [{
          "url": "...",
          "caption": "..."
          }]
      }
      ]
}
```

This is the description of a paper:
Paper is a scientific article that is associated with the following fields:

- *id*: Int;
- *title*: String;
- *abstract*: that is the summary of the paper: String;

- *authors*:
  - *name*: String;
  - *organization*: String;
  - *email*: that is a personal contact of the author: String;
  - *bio*: a short introduction written by the author (it is added from other sources): String;
- *FoS (Field of Study)*:
  - *name*: String;
  - *w*: that is the weight of the fields of study: Float;
- *Publication Details*:
  - *year*: that corresponds to the publication year: Int;
  - *page_start* and *page_end*: that are the starting and the ending point of the collection from which the paper was extracted: Int;
  - *doc_type*: that is the type of the paper, and can assume 3 values: *Journal*, *Conference*, *Patent*: String;
  - *publisher*: String;
  - *volume*: that corresponds to the n-th published collection: Int;
  - *issue*: that corresponds to the m-th part of the volume: Int;
  - *doi*: that is the Digital Object Identifier: String;
  - *n_citation*: that is the number of citations: Int;
- *Venue*: that is the collection from which the paper was extracted, with the attributes:
  - *id*: String;
  - *raw*: that is the name of the collection: String;
- *References*:
  - *id*: String;

- *Sections*: every section has an id, a title and a content (textual), it could have some subsections too (with the same structure of a section). Furthermore every section has one or more images. Every image has an url and a caption:
  - *id*: Int;
  - *title*: String;
  - *text*: String;
  - *subsections*:
    - *id*: Int;
    - *title*: Int;
    - *text*: String;
  - *figures*:
    - *url*: String;
    - *caption*: String;

2.2.1. Data Pre-Processing

In this part of the project we used as a starting point the same data set used in the first delivery, which can be downloaded *here*. However, the pre-processing this time was slightly different. Roughly the same operations of the first delivery were performed to clean and reduce the size of the data set. Those operations are performed with the script **dataset_preprocessing.py**, which can be found in the *scripts* folder of the *mongodb* section of the repository. It is worth to notice that such script does not split the original data set into multiple ones like in the first delivery. Instead, a single data set is kept.

2.2.2. Data Completion

We decided to avoid working with synthetic data (i.e. generate random meaningless data) and instead we decided to add to our dataset the missing fields by retrieving data from other data sets, trying to be as accurate as possible. The missing fields w.r.t. the description in section 2.2 were:

- *authors.email*
- *authors.bio*

- *sections*

We generated the email of every author and we inserted their bio, that were taken from Kaggle's Goodread-Authors data set. This is performed with the script **update_author.py**.

Furthermore we added the *Sections* part, using a Twitter dataset, that contains millions of tweets in different languages. For every paper we generated randomly a number of sections in the range from 1 to 3 and for each one from 0 to 2 subsections. For the title we used the text of one tweet and for the content the text of four tweets.

Then for the figures we used another data set (Train-GCC-training.tsv from the Google Conceptual Captions official website) in order to take the *image.url* and the *image.caption* fields. Every section contains a random number of figures between 1 and 3.

Those two operations above are executed in the notebook **sections_preprocessing.py**, that can be found in the *mongodb* section of the GitHub repository.

The final data set consists of 8333 documents.

To obtain the final data set starting from the downloaded one, run the scripts in this order:

1. **dataset_preprocessing.py**
2. **deprecated_references.py**
3. **bio_preprocessing.py**
4. **update_author.py**
5. **sections_preprocessing.py**

2.3. MongoDB

2.3.1. Data Upload

To import the data into a MongoDB collection, we rely on another Python script that exploits the PyMongo library, the official MongoDB driver for Python. We used the script **import_data.py**, that can be found in the *scripts* folder of the *mongodb* section of the GitHub repository. The script dumps the whole data set into a MongoDB collection, while preserving its complex and nested structure. It also performs some pre-processing on the *id* and on the *references* fields:

- The *id* field is renamed to *_id* in order to be used as an index inside of the database
- Both the *_id* field and the *references* field are converted to an ObjectId.

Once the correct connection parameters and the correct file path have been specified, it is enough to run the script with Python to import the data. We will use *papers* as the name of the collection.

2.3.2. Document Example

Below we can see the general structure of a document, with the help of MongoDB Compass.

```
  _id: ObjectId('101421652000000000000000')
  title: "The influence of query interface design on decision-making performance"
> authors: Array
> venue: Object
  year: 2003
  n_citation: 139
  page_start: 397
  page_end: 423
  doc_type: "Journal"
  publisher: "Society for Information Management and The Management Information Syst…"
  volume: 27
  issue: 3
> fos: Array
  doi: "10.2307/30036539"
> references: Array
  abstract: "Managers in modern organizations are confronted with ever-increasing v…"
> sections: Array
```

Figure 2.1: General structure

We will now expand the structure of the fields with a complex type.
The *authors* field is an array of sub-documents in which every element represents an author that worked on the paper.


```
authors: Array
  0: Object
     name: "Luke Olsen"
     id: "2142664686"
     org: "Department of Computer Science, University of Calgary, Calgary, AB, Ca…"
     email: "lukeolsen@mit.edu"
     bio: "Lucie Dufresne was born in 1951 in Trois-Rivières between two rivers: …"
  1: Object
     name: "Faramarz F. Samavati"
     id: "1821145341"
     org: "Department of Computer Science, University of Calgary, Calgary, AB, Ca…"
     email: "faramarzf.samavati@hotmail.com"
     bio: "Ramón González Férriz es editor y periodista. Actualmente, es columnis…"
  2: Object
     name: "Mario Costa Sousa"
     id: "2105740368"
     org: "Department of Computer Science, University of Calgary, Calgary, AB, Ca…"
     email: "mariocostasousa@mit.edu"
     bio: "Maja Ilisch, geboren 1975 in Dortmund, studierte Öffentliches Biblioth…"
  3: Object
     name: "Joaquim A. Jorge"
     id: "2120678171"
     org: "Departamento de Engenharia Informática, Instituto Superior Técnico, Li…"
     email: "joaquima.jorge@gmail.com"
     bio: "Author of TINCTURE (<a target="_blank" href="http://www.tincturestory.…"
```


Figure 2.2: Structure of the *authors* field

The *venue* field is a sub-document with two internal fields.


```
venue: Object
   raw: "Computers & Graphics"
   id: "94821547"
```


Figure 2.3: Structure of the *venue* field

The *fos* field is another array of sub-documents. Each element corresponds to one of the topics covered by the paper.

```
fos: Array
  0: Object
     name: "Computer vision"
     w: 0.427553326
  1: Object
     name: "Artificial intelligence"
     w: 0
  2: Object
     name: "Sketch recognition"
     w: 0.6052215
  3: Object
     name: "Geometric modeling"
     w: 0.48624804600000004
  4: Object
     name: "Human-computer interaction"
     w: 0.4427773
```

Figure 2.4: Structure of the *fos* field

The *references* field is implemented just as an array of ObjectIds.

```
references: Array
  0: ObjectId('175816726800000000000000')
  1: ObjectId('191857022600000000000000')
  2: ObjectId('209868554100000000000000')
  3: ObjectId('211598118400000000000000')
  4: ObjectId('212745062100000000000000')
  5: ObjectId('213274786700000000000000')
```

Figure 2.5: Structure of the *references* field

Lastly, the *sections* field is an array of sub-documents. Each element of such array corresponds to one section of the paper. Every section can contain another array of sub-documents in the *subsections* field and another array of sub-documents in the *figures* field.


```
sections: Array
  0: Object
    id: 1
    title: "İlk önce kendi gücünün bilgisine sahip olmalısın; ikincisi, meydan oku…"
    text: "Soyumuz soylansın, Boyumuz Boylansın.. Al Yıldızlı bayrağımız KUDÜSTE,…"
    subsections: Array
    figures: Array
      0: Object
        url: "http://lh6.ggpht.com/-IvRtNLNcG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/…"
        caption: "a very typical bus station    pop artist attends the 3rd annual at gue…"
      1: Object
        url: "http://lh6.ggpht.com/-IvRtNLNcG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/…"
        caption: "a very typical bus station    illustration of a map , its flag and a c…"
  1: Object
    id: 2
    title: "Gurbete düşmüş bir insan, ne denIi varIık içinde bir yaşam sürüyor oIs…"
    text: "Türk milleti vatanını koruyan ordumuzun yanındadır. #BarışPınarıViyadü…"
    subsections: Array
      0: Object
        id: 1
        title: "Hep ölerek çoğaldık… Biz Oğuzun erleri. #DavanınGücü"
        text: "Bizim milliyetçiliğimiz ayırıcı değil birleştirici, çatışmacı değil ba…"
      1: Object
        id: 2
        title: "“YPG silah ve malzemeleri bırakıp çekilsin, bu gece harekatı durduralı…"
        text: "RT @Enesovvic: Ya siz kimi kimin toprağından kovuyorsunuz?  Burası biz…"
    figures: Array
      0: Object
        url: "http://lh6.ggpht.com/-IvRtNLNcG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/…"
        caption: "a very typical bus station    rock artist performs on stage at awards …"
```


Figure 2.6: Structure of the *sections* field

2.3.3. Join Operation

To get a temporary collection in which every document also contains an array with inside all the other documents it references, we can use the *$lookup* operator.

```
db.papers.aggregate([
   {
     $lookup:
       {
         from: "papers",
         localField: "references",
         foreignField: "_id",
         as: "refs"
       }
  }
])
```

The referenced documents will be contained in the *refs* field.

2.3.4. Queries

In this section queries with a different level of complexity are presented, with a brief description and a figure that shows their results. Notice that, for some queries, that return several documents, the result is only partially shown.

1. **Find papers of 2006 with issue equal to 3 and volume greater or equal to 5. Then show just the list of authors with their name and organization, and also the venue of the paper. Limit the result to 2**:

```
db.papers.find(
   {"year": 2006, "volume": {"$gte": 5}, "issue": 3},
   {"authors.name":1, "authors.org":1, "venue":1}
).limit(2)
```

(nReturned: 2, executionTimeMillis: 57, totalDocsExamined: 382)

```
{ _id: ObjectId("168383441500000000000000"),
  authors:
   [ { name: 'A.D. Murugan',
       org: 'Dept. of Electr. & Comput. Eng., Ohio State Univ., Columbus, OH, USA' },
     { name: 'H. El Gamal',
       org: 'Dept. of Electr. & Comput. Eng., Ohio State Univ., Columbus, OH, USA' },
     { name: 'Mohamed Oussama Damen', org: 'University of Waterloo' },
     { name: 'Giuseppe Caire', org: 'Institut Eurecom' } ],
  venue:
   { raw: 'IEEE Transactions on Information Theory',
     id: '4502562' } }
{ _id: ObjectId("196436990000000000000000"),
  authors:
   [ { name: 'Kenneth C. Barr',
       org: 'MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA' },
     { name: 'Krste Asanovic',
       org: 'MIT Computer Science and Artificial Intelligence Laboratory, Cambridge, MA' } ],
  venue: { raw: 'ACM Transactions on Computer Systems', id: '193109227' } }
```


2. **Find 3 papers that were cited more than 500 times, published since 2015 in a Journal. Of them we retrieve only: title, publication year, number of citations, publisher, and doi**:

```
db.papers.find(
    {"n_citation": {"$gt": 500}, "year": {"$gte": 2015}, "doc_type": "Journal"},
    {"title": 1, "year": 1, "n_citation": 1, "publisher": 1, "doi": 1}
).limit(3)
```

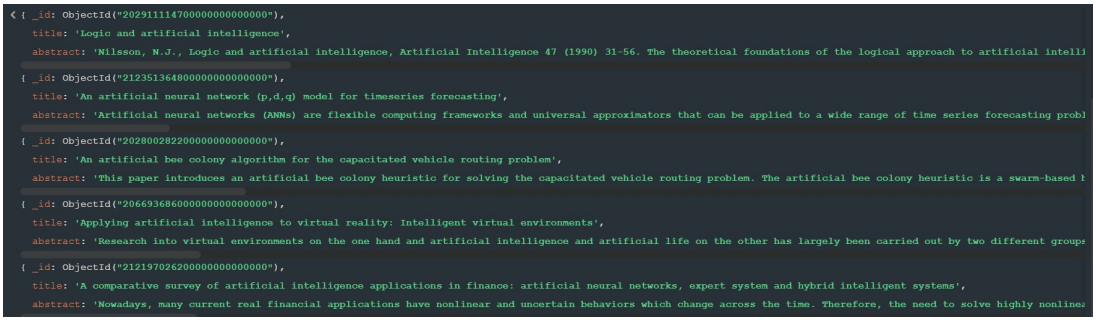
(nReturned: 3, executionTimeMillis: 3, totalDocsExamined: 326)


```
{ _id: ObjectId("161299778400000000000000"),
  title: 'ORB-SLAM: A Versatile and Accurate Monocular SLAM System',
  year: 2015,
  n_citation: 619,
  publisher: 'IEEE',
  doi: '10.1109/TRO.2015.2463671' }
{ _id: ObjectId("188518597100000000000000"),
  title: 'Image Super-Resolution Using Deep Convolutional Networks',
  year: 2016,
  n_citation: 641,
  publisher: 'IEEE',
  doi: '10.1109/TPAMI.2015.2439281' }
{ _id: ObjectId("191065790500000000000000"),
  title: 'SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation',
  year: 2017,
  n_citation: 551,
  publisher: 'IEEE',
  doi: '10.17863/CAM.17966' }
```

3. **Find 5 papers that contain the word "artificial" in their abstract.**
   To perform this query an index of type *text*, on the field *abstract* was previously created.

```
db.papers.find(
    {"$text": {"$search": "artificial"}},
    {"_id": 1, "title": 1, "abstract": 1}
).limit(5)
```

(nReturned: 5, executionTimeMillis: 9, totalDocsExamined: 5)



4. **Retrieve the sorted average number of pages, per doc_type, of papers from a year after 2000, or with a number of citation greater or equal to 100**:

```
db.papers.aggregate([
  {"$match": {"$or": [{"n_citation": {"$gte": 100}}, {"year":
  ↪ {"$gte": 2000}}]} },
  {
  "$group": {
    "_id": "$doc_type",
    "averageNumberPages": { "$avg": {"$subtract":
  ↪ ["$page_end","$page_start"]}}
  }},
  {
  "$sort": {"averageNumberPages": -1}}
])
```

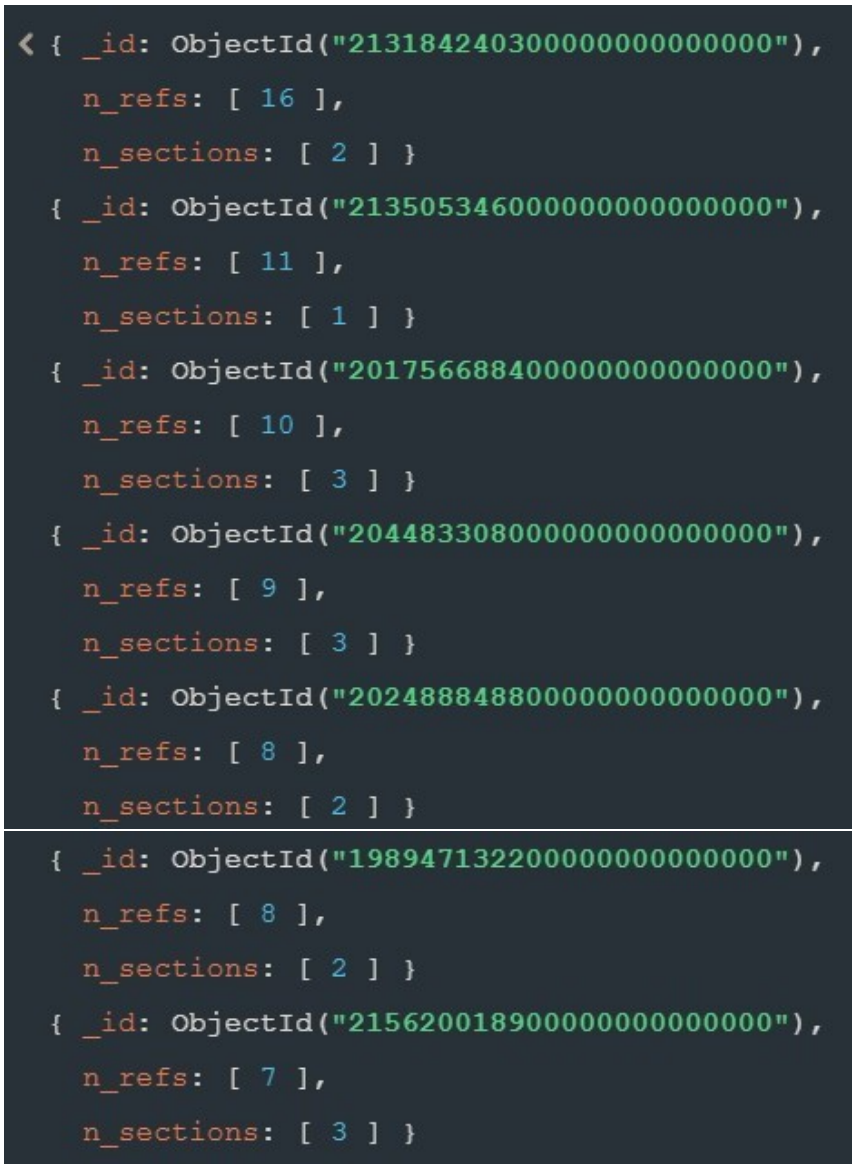
(nReturned: 3, executionTimeMillis: 195, totalDocsExamined: 8333)

```
{ _id: 'Patent', averageNumberPages: 23.428571428571427 }
{ _id: 'Journal', averageNumberPages: 23.040044649086088 }
{ _id: 'Conference', averageNumberPages: 16.00086281276963 }
```

5. **Find the 7 papers that contain the most references, among the papers that cover a very relevant field of study (weight >= 0.7), and that were presented in a Conference**:

```
db.papers.aggregate([
  {"$match": { "$and": [{"fos.w": {"$gte": 0.7}}, {"doc_type":
  ↪  "Conference"}] }},
  {
  "$group": {
    "_id": "$_id",
    "n_refs": {"$push": {"$size": "$references"}},
    "n_sections": {"$push": {"$size": "$sections"}}
  }},
  {"$sort": {"n_refs": -1}},
  {"$limit": 7}
])
```

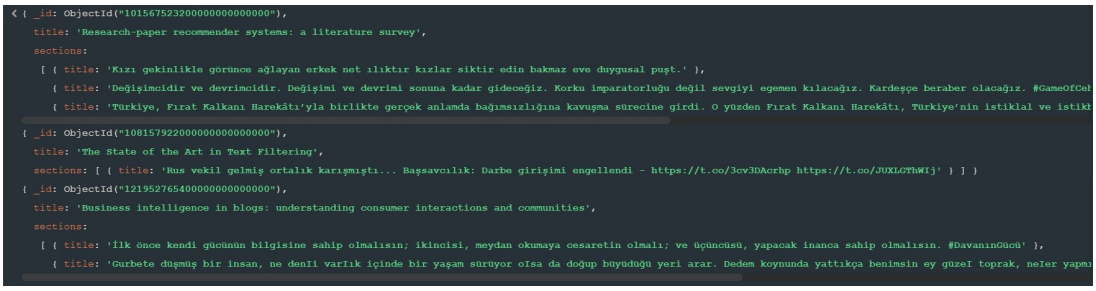
(nReturned: 7, executionTimeMillis: 14, totalDocsExamined: 8333)



6. **Find the papers whose fos (field of studies) is "Computer engineering" or "Computer science", and that have at least one subsection**:

```
db.papers.find(
    {"$and":[ {"$or": [{"fos.name": "Computer engineering"},
    {"fos.name": "Computer science"}]}, {"sections.subsections":
    {"$exists": true}} ] },
    {"title":1, "sections.title":1 }
)
```

(nReturned: 1789, executionTimeMillis: 31, totalDocsExamined: 8333)



7. **Find 10 papers with author affiliated with "IEEE", that have 3 sections and at least a figure's caption containing the word "bus":**

```
db.papers.find(
    {"$and": [{"authors": {"$elemMatch": {"org": "IEEE"}}},
     {"sections": {"$elemMatch": {"figures.caption": {"$regex":
     /bus/}}}}, {"sections": {"$size": 3}}]},
    {"title": 1, "year": 1, "n_citation": 1, "publisher": 1, "doi":
     1, "sections": 1}
).limit(10)
```

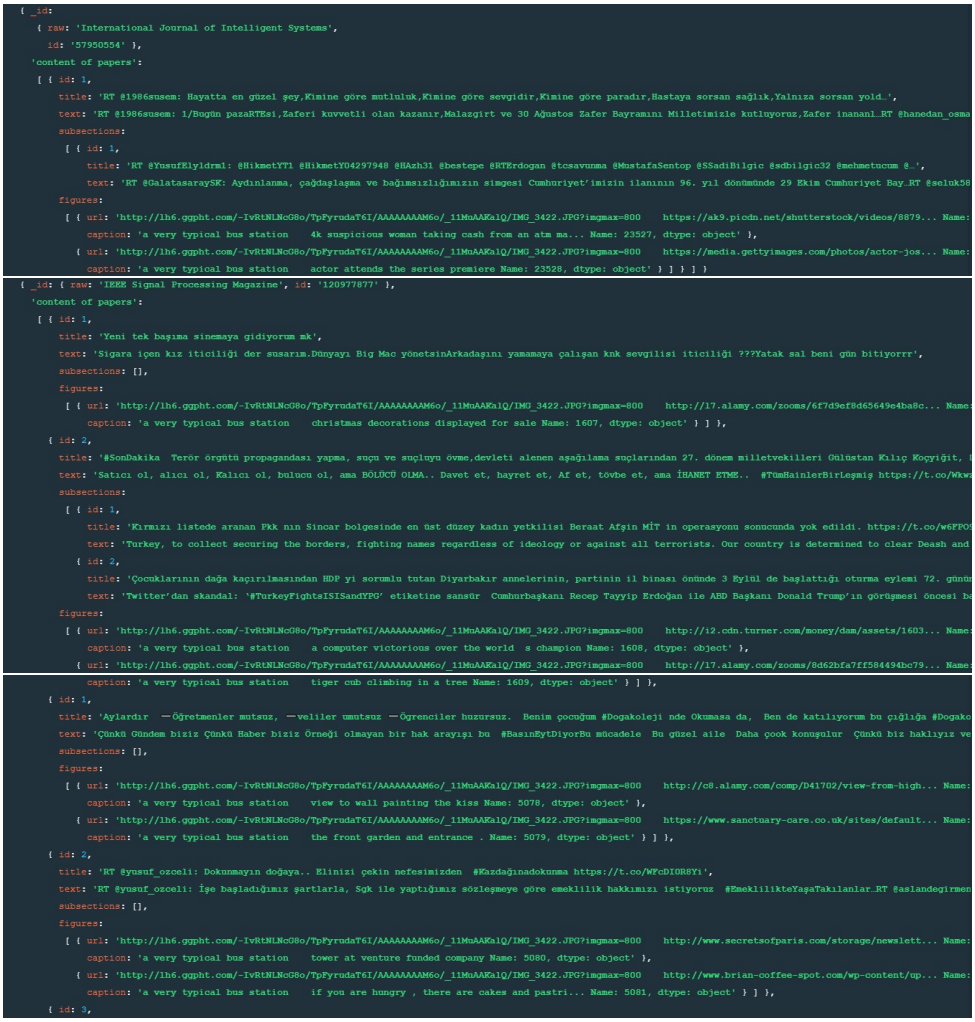
(nReturned: 10, executionTimeMillis: 21, totalDocsExamined: 7247)

8. **Retrieve for each venue the content of the papers, with more than 500 citations, and volume greater than 12, published in it.**

   This is done by first matching the conditions stated above, then unwinding on the sub-document "sections". In the end grouping by venue, and retrieving the sections, we obtain the following results (for each venue, an array with the contents collected from all the papers published in it):

```
db.papers.aggregate([
  {"$match": {"$and": [{"n_citation": {"$gte": 500}}, {"volume":
  ↪  {"$gt": 12}}]}},
  {"$unwind": {"path": "$sections"}},
  {"$group": {"_id": "$venue", "content of papers": {"$push":
  ↪  "$sections"}}}
])
```

(nReturned: 923, executionTimeMillis: 126, totalDocsExamined: 8333)



9. **Find the number of papers of the first 10 authors, affiliated to an University, that wrote most papers in Journals after year 2000.**

   This is done by first matching the conditions, and unwinding on the sub-documents "authors". Then by grouping by *authors.id*, the number of papers for each author can be computed. In the end the results are sorted, and the first 10 are shown.

```
db.papers.aggregate([
  {"$match":{ "$and": [{"doc_type":"Journal"}, {"year": {"$gt": 2000}}, {"authors.org": {"$regex": /University/}}]}},
  {"$unwind": {"path": "$authors"}},
  {"$group": {"_id": "$authors.id", "n_papers": {"$sum":1}}},
  {"$sort":{"n_papers":-1}},
  {"$limit": 10}
])
```

   (nReturned: 10, executionTimeMillis: 256, totalDocsExamined: 8333)

10. **Find the referenced papers whose *fos* (field of studies) is "Data mining" and whose volume is 3, and that have at least one section:**

```
db.papers.aggregate([
  { "$match": { "$and": [ {"fos.name": "Data mining"}, {"volume": 3}, {"sections": {"$exists": true} } ] } },
  {
    "$lookup":
      {
        "from": "Project2",
        "localField": "references",
        "foreignField": "_id",
```

```
9             "as": "refs"
10          }
11    }
12 ])
```

(nReturned: 24, executionTimeMillis: 37, totalDocsExamined: 8333)

11. **Find papers that are referenced by papers regarding machine learning, written by an IEEE author or published by "IEEE Computer Society" after 2010. Of the referenced papers retrieved with the join, show id, title and year of the ones published before 2005**:

```
db.papers.aggregate([
  { "$match": { "$and": [{"year": {"$gte": 2010}}, {"$or":
  ↪  [{"authors.org": "IEEE"}, {"publisher": "IEEE Computer
  ↪  Society"}]}, {"fos.name": "Machine learning"}]} },
  {
    "$lookup":
      {
        "from": "papersCollection",
        "localField": "references",
        "foreignField": "_id",
        "pipeline": [ {"$match": { "$expr": {"$lte": ["$year",
  ↪  2005]}}}, {"$project": {"_id": 1, "title": 1, "year": 1}}],
        "as": "refs"
      }
  }
])
```

(nReturned: 8, executionTimeMillis: 79, totalDocsExamined: 8333)


```
{ _id: ObjectId("19754367310000000000000"),
  title: 'Meta-Analysis of the First Facial Expression Recognition Challenge',
  authors:
   [ { name: 'Michel F. Valstar',
       id: '2040737520',
       org: 'Imperial Coll. London, London, UK',
       email: 'michelf.valstar@yahoo.com',
       bio: 'Geneviève Zubrzycki is Associate Professor of Sociology, Director of the Copernicus Program in Polish Studies, and Faculty Associate at the Frankel Center for Judai
     { name: 'Marc Mehu',
       id: '1723161209',
       org: 'Swiss Center for Affective Sci., Univ. of Geneva, Geneva, Switzerland',
       email: 'marcmehu@polimi.it',
       bio: 'Anglo-Welsh writer born to an army family in what was then Ceylon. <br /><br />After retiring from the army he adopted Wales as home and began to write of the count
     { name: 'Bihan Jiang',
       id: '2121798645',
       org: 'Dept. of Comput., Imperial Coll. London, London, UK',
       email: 'bihanjiang@gmail.com',
       bio: 'Over the years, I have written and edited a variety of books, magazines, Web sites, newspapers, and other publications, as well as authoring a number of non-fictior
     { name: 'Maja Pantic',
       id: '2085767126',
       org: 'Dept. of Comput., Imperial Coll. London, London, UK',
       email: 'majapantic@fastmail.com',
       bio: 'Munier Chowdhury (Bangla: মুনীর চৌধুরী), born in 1925 at Manikganj, Dhaka, hailed from Noakhali, was a Bangladeshi educationist, playwright, literary critic and poli
     { name: 'Klaus R. Scherer',
       id: '2311634644',
       org: 'Swiss Center for Affective Sci., Univ. of Geneva, Geneva, Switzerland',
       email: 'klausr.scherer@outlook.com',
       bio: 'Buddhadeva Bose (also spelt <a href="https://www.goodreads.com/search/search?q=Buddhadeb%20Bosu" title="Buddhadeb Bosu" rel="nofollow">Buddhadeb Bosu</a>) (Bengali:
  venue: { raw: 'systems man and cybernetics', id: '1170695740' },
  year: 2012,
  n_citation: 140,
  page_start: 966,
  page_end: 979,
  doc_type: 'Conference',
  publisher: 'IEEE Computer Society',
  volume: 42,
  issue: 4,
  fos:
   [ { name: 'Emotion recognition', w: 0 },
     { name: 'Machine learning', w: 0.427016944 },
     { name: 'Artificial intelligence', w: 0 },
     { name: 'Level playing field', w: 0.422916442 },
     { name: 'Standardization', w: 0.5232622 } ],
  doi: '10.1109/TSMCB.2012.2200675',
  references:
   [ ObjectId("20252165710000000000000"),
     ObjectId("20337730550000000000000"),
     ObjectId("20534322630000000000000"),
     ObjectId("20905184100000000000000"),
     ObjectId("21046803540000000000000"),
     ObjectId("21453104920000000000000"),
     ObjectId("21467806130000000000000"),
     ObjectId("21538226850000000000000"),
     ObjectId("21565031930000000000000"),
     ObjectId("21590172310000000000000"),
     ObjectId("21633528480000000000000") ],
  abstract: 'Automatic facial expression recognition has been an active topic in computer science for over two decades, in particular facial action coding system action unit (AU
  sections:
   [ { id: 1,
       title: 'RT @ufuk_durukan: Söz verdiğinde sözünün Arkasında Duran Tek Adam Başkan Erdoğan. https://t.co/MmFxjIwPfq',
       text: 'RT @goncalar07: Biz ne kadarız kaç yüz yada bin yoksa milyon hadin o zaman paylaşın pr destekli 20 milyonsa bu onu geçmeli @karabekiroglu_s_RT @AvErcanEZGN: Bir dü
       subsections:
        [ { id: 1,
            title: 'RT @E9ZoRqZtcBataiZAYmI5Hy+PGq8JvNo6Pgkgkh37HtM=: T A G I M I Z 👇👇👇👇  #ABOsuzEYTKatıksızEkmektir  23 Eylül günü başlattığımız tag çalışmamız bugün ve g
            text: 'RT @Eyt_MADUR: . Eyyy @TBMMresmi kapınıza alacaklınız dayandı..  Siz borçlu olarak ne yapmayı düşünüyorsunuz .?  #EYTliyizMeclisteyiz  @Akp_RT @Kadri06945977:
          { id: 2,
            title: 'RT @E9ZoRqZtcBataiZAYmI5Hy+PGq8JvNo6Pgkgkh37HtM=: @diskinsesi Teşekkürler @diskinsesi  #EYTliyiz@Eylüldeankaradayız Sizleri de aramızda yanımızda görmek bizi
            text: 'RT @ArzuLast: SOSYAL GÜVENLİK SİSTEMİ  Sosyal Olmayan İçinde Güven Kalmayan  Milyonlarca Mağdur Yaratan Kötü Bir Sistem...  Millet Burada Y_RT @burus_huseyin:
       figures:
        [ { url: 'http://lh6.ggpht.com/-IvRtNLNoG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/IMG_3422.JPG?imgmax=800    https://ssl.c.photoshelter.com/img-get/I0000Uc... Name: 2824,
            caption: 'a very typical bus station    fly fishing on the east branch . Name: 2824, dtype: object' },
          { url: 'http://lh6.ggpht.com/-IvRtNLNoG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/IMG_3422.JPG?imgmax=800    https://i.pinimg.com/736x/33/22/3e/33223e4ff3e... Name: 2825,
            caption: 'a very typical bus station    the sun may not always shine but the people do . Name: 2825, dtype: object' },
          { url: 'http://lh6.ggpht.com/-IvRtNLNoG8o/TpFyrudaT6I/AAAAAAAAM6o/_11MuAAKalQ/IMG_3422.JPG?imgmax=800    https://media.gettyimages.com/photos/jeff-brid... Name: 2826,
            caption: 'a very typical bus station    actor arrives at the world premiere held . Name: 2826, dtype: object' } ] },
```

2.3.5. Creation/Update Commands

1. **Deletion of one paper from the database**:

```
db.papers.deleteOne(
    { "_id": ObjectId("101421652000000000000000") }
)
```

2. **Deletion of some papers, based on some conditions**:

```
db.papers.deleteMany(
  {"$and" : [ {"n_citation" : {"$lt": 120} }, {"year": {"$lte":
  ↪  2005} } ] }
)
```

3. **Insertion of a new paper**:

```
db.papers.insertOne({
    "title": "The influence of query interface design on
    ↪  decision-making performance",
    "authors": [
      {
        "name": "Cheri Speier",
        "id": "1973614237",
        "org": "Michigan State University, East Lansing, MI#TAB#",
        "email": "cherispeier@polimi.it",
```

```
      "bio": "Dr. Cheri Speier-Pero is the Ernst & Young Professor of Accounting and Information Systems, faculty director of the MS in Business Analytics program, and interim chairperson of the Department of Supply Chain Management in the Eli Broad College of Business. She joined the Broad College faculty in 1998, and has since served as associate dean for MBA/MS Programs, led executive development courses, and served on many different committees at the university, college- and department-levels."
    },
    {
      "name": "Greta L. Polites",
      "id": "2305721212",
      "org": "School of Management, Bucknell University, Lewisburg, PA#TAB#",
      "email": "gretal.polites@123mail.org",
      "bio": "Dr. Polites joined the Kent State faculty in Fall 2012. She earned her B.S., M.S., and M.B.A. degrees from the University of South Florida, and completed her Ph.D. in Business Administration at the University of Georgia.[. . .] She has published two papers in the field of invertebrate paleontology, and has two fossil mollusk species (Attiliosa gretae and Opalia politesae) named after her."
    }
  ],
  "venue": {
    "raw": "Management Information Systems Quarterly",
    "id": "57293258"
  },
  "year": 2003,
  "n_citation": 139,
  "page_start": 397,
  "page_end": 423,
  "doc_type": "Journal",
  "publisher": "Society for Information Management and The Management Information Systems Research Center",
  "volume": 27,
  "issue": 3,
```

```
    "fos": [
      {
        "name": "Decision-making",
        "w": 0.4957332
      },
      {
        "name": "Knowledge management",
        "w": 0.46034035100000004
      },
      {
        "name": "Workload",
        "w": 0.5390811560000001
      },
      {
        "name": "Interface design",
        "w": 0
      },
      {
        "name": "Information system",
        "w": 0.5624888
      }
    ],
    "doi": "10.2307/30036539",
    "references": [
      ObjectId("151626165300000000000000"),
      ObjectId("197873803500000000000000")
    ],
```

```
58      "abstract": "Managers in modern organizations are confronted
    ↪ with ever-increasing volumes of information that they must
    ↪ evaluate when making a decision. Data warehousing and data
    ↪ mining technologies have given managers a number of valuable
    ↪ tools that can help them store, retrieve, and analyze
    ↪ information contained in large databases; however, maximizing
    ↪ user performance with these tools remains a challenge for
    ↪ information systems professionals. [. . .] These results have
    ↪ important implications for the design of managerial
    ↪ decision-making systems, particularly in complex decision-making
    ↪ environments.",
59      "sections": [
60        {
61          "id": 1,
62          "title": "Introduction,
63          "text": "Automated decision-making (ADM) is no longer
    ↪ science fiction and systems are now making decisions that were
    ↪ traditionally made by humans (Robert, Pierce, Marquis, Kim, &
    ↪ Alahmad, 2020). Algorithms match passengers for a shared ride
    ↪ and plan drivers' routes (Möhlmann & Zalmanson, 2017). In Hong
    ↪ Kong, an algorithm organizes the underground's maintenance
    ↪ schedule and assigns repair jobs to service technicians (Hodson,
    ↪ 2014).",
64          "subsections": [],
65          "figures": [
66            {
67              "url": "http://lh6.ggpht.com/IMG_3422.JPG?imgmax=800",
68              "caption": "Chat interaction from participants' point of
    ↪ view"
69            },
70            {
71              "url": "http://lh6.ggpht.com/IMG_3422.JPG?imgmax=800",
72              "caption": "Interview with the system after round one"
73            },
74            {
75              "url": "http://lh6.ggpht.com//IMG_3422.JPG?imgmax=800",
```

```
            "caption": "Pictures of the anthropomorphism manipulation"
          }
        ]
      }
    ]
  })
```

4. **Insertion of more papers at once**:

```
db.papers.insertMany(
  [{
    "title": "Shackled to [...] acceptance",
    "authors": [
      {
        "name": "Elena Karahanna",
        "id": "270263451",
        "org": "University of Georgia, Athens",
        "email": "elenakarahanna@liberomail.com",
        "bio": "Education: PhD, MIS, University of Minnesota, 1993.[...]"
      }
    ],
    "venue": {
      "raw": "Management Information Systems Quarterly",
      "id": "57293258"
    },
    "year": 2012,
    "n_citation": 215,
    "page_start": 21,
    "page_end": 42,
    "doc_type": "Journal",
    "publisher": "Society for Information Management",
    "volume": 36,
    "issue": 1,
    "fos": [
      {
        "name": "Management science",
```

```
      "w": 0.3945593
    }
  ],
  "doi": "10.2307/41410404",
  "references": [
    ObjectId("175816726800000000000000"),
    ObjectId("191857022600000000000000"),
    ObjectId("209868554100000000000000")
  ],
  "abstract": "Given that adoption of a new system often implies
↪ fully or partly replacing [...]",
  "sections": [
    {
      "id": 1,
      "title": "Introduction",
      "text": "Warfarin, [...]",
      "subsections": [
        {
          "id": 1,
          "title": "Warfarin dose adjustment",
          "text": "Warfarin dose adjustment [...]"
        },
        {
          "id": 2,
          "title": "Time in therapeutic range (TTR)",
          "text": "Time in therapeutic range (TTR) has been widely
↪ used. [...]"
        }
      ],
      "figures": [
        {
          "url": "http://lh6.ggpht.com/IMG_3422.JPG?imgmax=800",
          "caption": "PRISMA chart used for the selection of
↪ articles"
        },
        . . .
      ]
```

```
    },
    {
      "id": 2,
      "title": "Method",
      "subsections": [
        {
          "id": 1,
          "title": "Search Strategy",
          "text": "We conducted an extensive literature search in
↪ a very systematic way [...]"
        }
      ],
      "figures": [
        {
          "url": "http://lh6.ggpht.com//IMG_3422.JPG?imgmax=800",
          "caption": "Poor hypertension control or age-related
↪ frailty"
        }
      ]
    },
    {
      "id": 3,
      "title": "Result",
      "subsections": [
        {
          "id": 1,
          "title": "Selection Criteria",
          "text": "1) Age: There is clear evidence on the benefits
↪ of warfarin [...]"
        },
        {
          "id": 2,
          "title": "Data Extraction",
          "text": "Two reviewers (TR and NKJ1) independently
↪ extracted data [...]"
        }
          . . .
```

```
        ],
        "figures": [
          {
            "url": "http://lh6.ggpht.com/IMG_3422.JPG?imgmax=800",
            "caption": "Optimal anticoagulation"
          }
        ]
      }
    ]
  },
  {
    "title": "The State of the Art in Text Filtering",
    "authors": [
      {
        "name": "Douglas W. Oard",
        "id": "7916806",
        "org": "University of Maryland, College Park, MD 20742,
↪ U.S.A.",
        "email": "douglasw.oard@123mail.org",
        "bio": "Institute for Advanced Computer Studies.Information
↪ technology, user modeling, information retrieval. [...]"
      }
    ],
    "venue": {
      "raw": "User Modeling and User-adapted Interaction",
      "id": "160628929"
    },
    "year": 1997,
    "n_citation": 114,
    "page_start": 141,
    "page_end": 178,
    "doc_type": "Journal",
    "publisher": "Kluwer Academic Publishers",
    "volume": 7,
    "issue": 3,
    "fos": [
      {
```

```
        "name": "Machine learning",
        "w": 0.442438453
      },
      {
        "name": "Computer science",
        "w": 0.4239926
      },
    ],
    "doi": "10.1023/A:1008287121180",
    "references": [ ... ],
    "abstract": "This paper develops. [. . .] implications for
    ↪ future research on text filtering.",
    "sections": [
      {
        "id": 1,
        "title": "Introduction",
        "text": "Warfarin, a coumarin derivative oral anticoagulant
    ↪ [...]",
        "subsections": [
          {
            "id": 1,
            "title": "Warfarin dose adjustment",
            "text": "Warfarin dose adjustment is based on regular
    ↪ monitoring of international normalized ratio (INR). [...]"
          },
          {
            "id": 2,
            "title": "Time in therapeutic range (TTR)",
            "text": "Time in therapeutic range (TTR) has been widely
    ↪ used to measure the quality of INR control [...]"
          }
        ],
        "figures": [
          {
            "url": "http://lh6.ggpht.com/IMG_3422.JPG?imgmax=800",
            "caption": "PRISMA chart used for the selection of
    ↪ articles"
```

```
            }
          ]
        }
      ]
    }]
)
```

5. **Update of a paper, by modification of the title of a subsection**:

```
db.papers.updateOne(
  { "_id":  ObjectId("101567523200000000000000"), "authors.name":
  ↪  "Joeran Beel"  } ,
  { "$set": { "sections.0.subsections.0.title": "Artificial
  ↪  Intelligence: Machine Learning"} }
)
```

6. **Update some papers, based on some conditions, modifying starting and ending pages**:

```
db.papers.updateMany(
   { "doc_type":  "Conference", "year": {"$gte": 2000} } ,
   { "$set": { "page_start": 0 } },
   { "$set": { "page_end": {"$substract":
  ↪  ["$page_end","$page_start"] } } }
)
```

3 | Third Delivery

3.1. Introduction

This project aims at building an Information System that manages a data set containing different type of scientific articles that can be used for: clustering with network and side information, studying influence in the citation network, finding the most influential papers and topics, modeling analysis.

For the same problem described in Chapter 1 (First Delivery), we modeled a different system following the subsequent steps.
At first the data set was pre-processed to fit the correct structure requested by Spark. Then it was reduced in size and after that it was uploaded on Spark.
At the end of the project 10 queries and 5 creation/update commands were initiated with a different level of complexity that was checked within the performance time.

3.2. Data

The structure of the data is similar to the one used in the first delivery with 4 different entities: Paper, Author, FoS and Venue. The main difference between the data used for Neo4j and this one is given by the fact that in this new version some pre-processing is done to make the entities independent from the papers.

3.2.1. Data Pre-Processing

In this part of the project we used as a starting point the same data set used in the first delivery, which can be downloaded *here*. However, the pre-processing this time was slightly different. Roughly the same operations of the first delivery were performed to clean and reduce the size of the data set. Those operations are performed with the script **dataset_preprocessing.py**, which can be found in the *scripts* folder of the *spark* section

of the repository.

This pre-processing script splits the original data set into multiple ones, just like in the First Delivery. Thus, the splitting follows the structure of the ER diagram, ending up with a separate data set for the entities **Paper**, **Author**, **Venue** and **FoS**.

One additional step was performed before importing the data into Spark: we decided to make every entity independent from the others using unique ids. This was done to perform join operations between multiple DataFrames more easily. The **Paper**, **Author** and **Venue** data sets already had such ids. On the other hand, for the **FoS** data set we had to generate them.

In order for this to work, we had to insert into the **Paper** data set the ids of:

- The authors that worked on the paper
- The venue in which the paper was presented
- The fields of study that the paper covers

The script **spark_preprocessing.py** handles these reconstruction operations for the **Paper**, **Author** and **Venue** data sets, exploiting a grouping operation to drop duplicates.

On the other hand, the script **fos_preprocessing.py** handles the creation of unique ids and the reconstruction operations for the **FoS** data set, grouping together the FoS with the same name and computing an average of their corresponding weights.

The final data set consists of 8333 papers, 19445 authors, 726 venues and 6088 FoS.

To obtain the final data set starting from the downloaded one, run the scripts in this order:

1. **dataset_preprocessing.py**
2. **deprecated_references.py**
3. **spark_preprocessing.py**
4. **fos_preprocessing.py**

3.3. Spark

3.3.1. Data Schema

In this section, we describe the schema of the DataFrames that will be used to handle the data in Spark. The schemas are constructed using the classes *StructType* and *StructField* imported from *PySpark*. The data types of the field are imported from *PySpark* too.

Creation of the schema of the paper DataFrame:

```
schema_paper = StructType([
    StructField("id", StringType(), False),
    StructField("title", StringType(), False),
    StructField("year", IntegerType(), True),
    StructField("n_citation", IntegerType(), True),
    StructField("page_start", IntegerType(), True),
    StructField("page_end", IntegerType(), True),
    StructField("doc_type", StringType(), True),
    StructField("publisher", StringType(), True),
    StructField("volume", IntegerType(), True),
    StructField("issue", IntegerType(), True),
    StructField("doi", StringType(), True),
    StructField("references", StringType(), True),
    StructField("abstract", StringType(), True),
    StructField("venue_id", LongType(), False),
    StructField("authors_id", StringType(), False),
    StructField("fos_id", StringType(), False)
])
```

Creation of the schema of the author DataFrame:

```
schema_author = StructType([
    StructField("id", StringType(), False),
    StructField("name", StringType(), False),
    StructField("org", StringType(), True)
])
```

Creation of the schema of the venue DataFrame:

```
schema_venue = StructType([
    StructField("id", LongType(), False),
```

```
    StructField("name", StringType(), False),
])
```

Creation of the schema of the fos DataFrame:

```
schema_fos = StructType([
    StructField("id", StringType(), False),
    StructField("name", StringType(), False),
    StructField("weight", FloatType(), True)
])
```

3.3.2. Data Upload

In order to upload the data, the *PySpark* library is exploited once again. Given that the schemas have already been defined, we can populate the DataFrames by inserting the data sets in the CSV format using the *read.csv(path)* method of the SparkSession class, made available by *PySpark*.

Here we can see a working example with the author dataset:

```
author_df = spark.read.option("header", True).option("delimiter",
↪  ",").schema(schema_author).csv("author_spark.csv")
```

In a very similar way, we can import also the venue and the fos data sets. The paper data set, on the other hand, requires some further preprocessing due to its complex structure. In particular, we have to make sure to:

- Escape the *"* character when reading some textual fields like *title* and *abstract*, in order to parse correctly their corresponding string.
- Reconstruct the fields *references*, *authors_id* and *fos_id* so that they are correctly an array of strings instead of a single string.

The complete code to perform this preprocessing can be found in the **import_data_and_queries.ipynb** script in the *spark* section of the GitHub repository.

3.3.3. Queries

In this section queries with a different level of complexity are presented, with a brief description and a figure that shows their results. Notice that, for some queries, that return several items, the result is only partially shown.

The execution time of each query was measured using the magic command *%%timeit*: it is an IPython command that allows measuring the execution time of the cell where it's written. Clearly, it is not the best way to measure the performance of a database query, but it can still be useful since it can be used to measure the difference in time between the simple queries and the more complex ones.

Such queries can be found and executed in the **import_data_and_queries.ipynb** script in the *spark* section of the GitHub repository.

1. **Retrieve the papers which have at least Computer Science among their fos.**

```
1 paper_df.select(col("id"), col("title"), col("year"),
  ↪ col("n_citation"), col("publisher"), col("references"),
  ↪ col("abstract"), col("venue_id"), col("authors_id"),
  ↪ explode(col("fos_id"))) \
2 .withColumnRenamed("col", "fos_id") \
3 .withColumnRenamed("id", "paper_id") \
4 .join(fos_df, fos_df.id == col("fos_id"),
  ↪ "fullouter").filter(col("name") == "Computer science") \
5 .select(col("paper_id"), col("title"), col("year"),
  ↪ col("n_citation"), col("publisher"), col("references"),
  ↪ col("venue_id"), col("authors_id"), col("fos_id")) \
6 .show()
```

   Execution time: 829 ms ± 37.5 ms

paper_id	title	year	n_citation	publisher	references	venue_id	authors_id	fos_id
86453134	IDDQ testing: A r...	1992	169	Springer US	[]	200807567	[2112919840, 214...	100963
767067438	A review of the l...	2016	142	Elsevier	[1982564000, 198...	205292342	[2140778852]	100963
69854901	Survey A survey o...	2012	160	Elsevier North-Ho...	[2003992171, 200...	63392143	[2119852781, 318...	100963
639708223	Faster R-CNN: Tow...	2017	2586	IEEE	[1958328135, 203...	199944782	[2119543935, 216...	100963
433644524	Review: Intrusion...	2013	271	Academic Press Ltd.	[2007087405, 206...	30128005	[2132462014, 265...	100963
40650588	Search and pursui...	2011	197	Springer US	[2106518318, 211...	144091109	[2121390041, 219...	100963
2776316078	SMT-Based Bounded...	2012	128	IEEE	[1993836075, 205...	8351582	[2117275544, 261...	100963
2766000922	Adaptive Query Pr...	2007	232	Now Publishers Inc.	[2060883486, 209...	139685423	[2002742946, 535...	100963
2761239369	Achieving the Sec...	2011	242	IEEE	[]	4502562	[1978228342, 211...	100963
2668860018	Concept abduction...	2005	101	Elsevier	[1515520450, 159...	187709482	[2107847746, 477...	100963
2616840662	Effects of Relati...	2005	132	M. E. Sharpe, Inc.	[1516261653, 209...	9954729	[2633097997, 216...	100963
2616747538	The Datacenter as...	2013	140	Morgan & Claypool...	[2064359039, 212...	962203723	[2023575095, 270...	100963
2613214602	Parameter free bu...	1992	475	VLDB Endowment	[2044555065, 207...	1133523790	[1985207418, 237...	100963
2601109700	A Temporal Logic ...	2017	120	ACM	[2029408547, 214...	118992489	[2130070262]	100963
2469486851	A Privacy-Preserv...	2016	270	IEEE	[1979493600, 205...	61310614	[2146706295, 250...	100963
2442495973	Summarizing scien...	2002	382	MIT press	[1497212108, 204...	155526855	[1806051850, 212...	100963
2438667436	Partially observa...	2007	510	Academic Press Ltd.	[2158907787, 216...	91252481	[2143360327, 212...	100963
2430154064	Secure data aggre...	2009	263	Elsevier	[2125572812, 213...	63392143	[2123023843, 211...	100963
2428120229	Data gathering op...	2016	137	IEEE	[]	62238642	[2664807554, 211...	100963
2426038346	VoIP: A comprehen...	2009	111	Elsevier	[2148810932]	63392143	[2046179957, 200...	100963

2. **Retrieve papers that have more than 400 citations and of any IEEE publisher (i.e. the publisher must be "IEEE", or "IEEE" concatenated with other strings, for example "IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS"). Then take only the first 15 results.**

```
paper_df.filter((col("n_citation") >= 400) &
    (col("publisher").like("IEEE%"))) \
.limit(15) \
.select(col("id"), col("title"), col("n_citation"),
    col("publisher"), col("abstract")) \
.show()
```

Execution time: 273 ms ± 89.5 ms

id	title	n_citation	publisher	abstract
1510186039	Graspit! A versat...	521	IEEE	A robotic graspin...
1612997784	ORB-SLAM: A Versa...	619	IEEE	This paper presen...
1641498739	The Multimodal Br...	417	IEEE	In this paper we ...
1885185971	Image Super-Resol...	641	IEEE	We propose a deep...
1910657905	SegNet: A Deep Co...	551	IEEE	We present a nove...
1963932623	Label Consistent ...	469	IEEE	A label consisten...
1964402820	Regular and irreg...	992	IEEE	We propose a gene...
1964793896	OFDM Versus Filte...	598	IEEE	As of today, orth...
1965497192	Network-Induced C...	500	IEEE TRANSACTIONS...	Networked control...
1965665622	Ranging With Ultr...	536	IEEE	Over the coming d...
1966000877	Wireless Networks...	627	IEEE	Radio frequency (...
1970334960	Principles of Phy...	423	IEEE	This paper provid...
1971616954	Fast Keypoint Rec...	463	IEEE Computer Soc...	While feature poi...
1971875039	Intuitionistic Fu...	965	IEEE	An intuitionistic...
1974042113	RASL: Robust Alig...	503	IEEE	This paper studie...

3. **Retrieve the papers of conferences since 2003.**

```
paper_df.filter((col("id") \
  .isin(paper_df.filter(col("doc_type") == "Conference") \
```

```
    .select(collect_list("id")).collect()[0][0])) & (col("year") > 2003)) \
.select(col("id"), col("title"), col("year"), col("doc_type"), col("abstract"), col("venue_id"), col("authors_id"), col("fos_id")) \
.show(10)
```

Execution time: 2.08 s ± 137 ms

id	title	year	doc_type	abstract	venue_id	authors_id	fos_id
1485941238	Succinct suffix a...	2005	Conference	A succinet full-t...	1192804102	[2026327678, 213...	[103247, 101525,...
1515520450	Data integration ...	2004	Conference	Data integration ...	1134069326	[2781243587, 198...	[101294, 102665,...
1516360493	Global Progress i...	2008	Conference	A multiparty sess...	1145706541	[2035671728, 214...	[100963, 101552,...
1537747159	Data Clustering: ...	2008	Conference	The practice of c...	2755314191	[2162010601]	[100763, 105371,...
1580065766	Symmetry in 3D Ge...	2013	Conference	The concept of sy...	2754362256	[2136233650, 212...	[100970, 100270,...
1594489547	BnB-ADOPT: an asy...	2008	Conference	Distributed const...	1168671587	[2144475260, 193...	[100270, 103154,...
1597082186	PARIS: probabilis...	2011	Conference	One of the main c...	1133523790	[69603646, 17240...	[100963, 101294,...
1699577049	Declassification:...	2009	Conference	Computing systems...	2615919549	[313407794, 2227...	[104676, 102650,...
1742181809	Image-based mater...	2005	Conference	Photo editing sof...	1164321581	[2108391018, 247...	[100970, 100270,...
1760407951	TILT: transform i...	2010	Conference	In this paper, we...	1174644639	[2660899906, 270...	[100970, 100270,...

4. **Compute how many papers were published, for each year, on each venue, showing also the venue name. Display results ordered by year in descendent order.**

```
paper_df.join(venue_df, venue_df.id == paper_df.venue_id) \
.groupBy("year", "venue_id", "name") \
.agg(count("name").alias("n_publications")) \
.orderBy(desc("year")) \
.show(20, truncate = False)
```

Execution time: 614 ms ± 47.1 ms

year	venue_id	name	n_publications
2018	199944782	IEEE Transactions on Pattern Analysis and Machine Intelligence	1
2018	160107561	Siam Review	1
2017	61310614	IEEE Transactions on Information Forensics and Security	1
2017	134177497	IEEE Transactions on Fuzzy Systems	1
2017	76152103	IEEE Transactions on Systems, Man, and Cybernetics	2
2017	25538012	International Journal of Computer Vision	1
2017	7560371	Information Fusion	1
2017	193920097	Mathematical Programming	1
2017	199944782	IEEE Transactions on Pattern Analysis and Machine Intelligence	6
2017	110206669	Multimedia Tools and Applications	1
2017	115304631	IEEE Transactions on Image Processing	1
2017	51360982	Automatica	1
2017	118992489	Journal of the ACM	1
2017	161464388	Journal of Intelligent Manufacturing	1
2017	95999327	IEEE Systems Journal	1
2017	1131341566	international symposium on computer architecture	1
2017	62401924	Journal of the American Statistical Association	1
2017	42080949	IEEE Transactions on Neural Networks	2
2017	45693802	Neurocomputing	1
2016	1160032607	symposium on principles of programming languages	1

5. **Compute how many papers were published, for each year, by the IEEE and with a number of citations greater than 500.**

```
paper_df.filter((col("publisher") == "IEEE") & (col("n_citation") >
↪  500)) \
.groupBy("year", "doc_type") \
.agg(count("year").alias("n_years")) \
.orderBy(desc("year")) \
.show(20, truncate = False)
```

Execution time: 397 ms ± 79.7 ms

```
+----+----------+-------+
|year|doc_type  |n_years|
+----+----------+-------+
|2018|Journal   |1      |
|2017|Journal   |2      |
|2016|Journal   |2      |
|2015|Journal   |5      |
|2014|Journal   |14     |
|2013|Journal   |12     |
|2012|Conference|2      |
|2012|Journal   |11     |
|2011|Journal   |23     |
|2011|Conference|1      |
|2010|Journal   |18     |
|2010|Conference|1      |
|2009|Conference|2      |
|2009|Journal   |16     |
|2008|Journal   |14     |
|2008|Conference|2      |
|2007|Journal   |18     |
|2007|Conference|3      |
|2006|Journal   |14     |
|2005|Journal   |14     |
+----+----------+-------+
```

6. **Retrieve the publishers that have more than 100 papers stored in the database.**

```
paper_df.groupBy("publisher") \
.agg(count("id").alias("n_papers")) \
.filter(col("n_papers")>=100) \
.show(truncate = False)
```

Execution time: 429 ms ± 167 ms

```
+-------------------------------------------------------------------------------------+--------+
|publisher                                                                            |n_papers|
+-------------------------------------------------------------------------------------+--------+
|Society for Information Management and The Management Information Systems Research Center|105     |
|Elsevier                                                                             |565     |
|Elsevier Science Inc.                                                                |193     |
|Elsevier Science Publishers Ltd.                                                     |108     |
|ACM                                                                                  |1104    |
|Kluwer Academic Publishers                                                           |277     |
|Springer US                                                                          |247     |
|IEEE                                                                                 |2092    |
|Springer-Verlag                                                                      |163     |
|IEEE Press                                                                           |142     |
|IEEE Computer Society                                                                |433     |
|Elsevier Science Publishers B. V.                                                    |272     |
|Pergamon                                                                             |131     |
+-------------------------------------------------------------------------------------+--------+
```

7. **Retrieve the years in which the number of published paper of journals is greater than 150.**

```
paper_df.filter(col("doc_type") == "Journal") \
.groupBy("year") \
.agg(count("id").alias("n_papers")) \
.orderBy(asc("year")) \
.filter(col("n_papers") > 150) \
.show(truncate = False)
```

Execution time: 380 ms ± 87.2 ms

```
+----+--------+
|year|n_papers|
+----+--------+
|1998|158     |
|2000|187     |
|2001|181     |
|2002|231     |
|2003|309     |
|2004|296     |
|2005|421     |
|2006|422     |
|2007|464     |
|2008|485     |
|2009|505     |
|2010|559     |
|2011|574     |
|2012|478     |
|2013|466     |
|2014|323     |
|2015|188     |
+----+--------+
```

8. **Compute the total number of citations, for each publisher, of the papers that have a field of study with weight grater than 0.6.**

```
paper_df.select(col("id"), col("title"), col("year"),
    col("n_citation"), col("doi"), col("publisher"),
    explode(paper_df.fos_id)) \
.withColumnRenamed("col", "fos") \
.filter(col("fos") \
  .isin(fos_df.filter(col("weight") > 0.6) \
  .select(collect_list("id")) \
  .collect()[0][0])) \
.groupBy(col("publisher")) \
.sum("n_citation") \
.withColumnRenamed("sum(n_citation)", "total_citations") \
.show(20, truncate = False)
```

Execution time: 2.02 s ± 135 ms

publisher	total_citations
Cambridge University Press	374
捷頂科技有限公司	112
Wiley-Blackwell	478
John Wiley & Sons, Ltd.	345
Elsevier	2339
Kluwer Academic Publishers-Plenum Publishers	463
Decision Support Systems	136
Elsevier Science Inc.	678
MIT Press 238 Main St., Suite 500, Cambridge, MA 02142-1046USA journals-info@mit.edu	412
IEEE Computer Society Press	878
John Wiley & Sons, Inc.	248
John Wiley & Sons	155
Elsevier Science Publishers Ltd.	137
ACM	15492
Springer International Publishing	122
Academic Press, Inc.	642
North-Holland	943
Kluwer Academic Publishers	5407
Springer US	199
IEEE	11892

9. **Retrieve the authors whose organization is the Stanford University and that wrote a number of papers greater than 1.**

```
paper_df.select(col("id"), col("title"), col("year"),
    col("n_citation"), col("page_start"), col("page_end"),
    col("doc_type"), col("publisher"), col("volume"), col("issue"),
    col("doi"), col("references"), col("abstract"), col("venue_id"),
    explode(col("authors_id")), col("fos_id")) \
.withColumnRenamed("col", "authors_id") \
.withColumnRenamed("id", "paper_id") \
.join(author_df, author_df.id == col("authors_id"), "full") \
.filter(col("org") == "Stanford University") \
```

```
.groupBy("authors_id") \
.agg(count("paper_id").alias("n_papers")) \
.orderBy(asc("n_papers")) \
.filter(col("n_papers") > 1) \
.show(truncate = False)
```

Execution time: 837 ms ± 137 ms

```
+----------+--------+
|authors_id|n_papers|
+----------+--------+
|2149433985|2       |
|201828038 |4       |
|348630313 |4       |
+----------+--------+
```

10. **Retrieve the venues on which an organization has published 4 or more papers, that consist of more than 10 pages.**

```
paper_df.filter((col("page_end") - col("page_start") > 10)) \
.select(col("id").alias("paper_id"), col("venue_id"), col("volume"),
↪  explode(paper_df.authors_id).alias("author_id")) \
.join(author_df, author_df.id ==
↪  col("author_id")).drop("id").withColumnRenamed("name",
↪  "author_name") \
.join(venue_df, venue_df.id ==
↪  paper_df.venue_id).drop("id").withColumnRenamed("name",
↪  "venue_name") \
.groupBy(col("venue_id"), col("venue_name"), col("org")) \
.agg(count("paper_id").alias("n_papers")) \
.filter((col("n_papers") >= 4) & (col("org").isNotNull())) \
.sort(desc("n_papers")) \
.show()
```

Execution time: 1.05 s ± 91 ms

venue_id	venue_name	org	n_papers
134177497	IEEE Transactions...	Space Control & I...	9
134177497	IEEE Transactions...	Department of Mat...	7
1127352206	programming langu...	Microsoft Researc...	5
157921468	ACM Computing Sur...	Columbia Univ., N...	5
1152462849	acm special inter...	University of Ill...	5
168680287	IEEE Transactions...	Technion - Israel...	4
57293258	Management Inform...	Fox School of Bus...	4
90119964	ACM Transactions ...	Univ. of Maryland...	4
25538012	International Jou...	Stanford Universi...	4
63459445	IEEE Transactions...	Dept. of Electr. ...	4
199944782	IEEE Transactions...	Nat. Lab. of Patt...	4
414566	Pattern Recognition	Department of Com...	4
93787993	IEEE Transactions...	Department of Com...	4
8351582	IEEE Transactions...	Lane Dept. of Com...	4
80113298	Journal of the As...	University of Wol...	4
63392143	Computer Networks	Broadband and Wir...	4

3.3.4. Creation/Update Commands

Such commands can be found and executed in the **import_data_and_queries.ipynb** script in the *spark* section of the GitHub repository.

1. **Drop the column year from the paper data frame.**:

```
paper_df.drop(paper_df.year) \
    .show()
```

Execution time: 195 ms ± 39.7 ms

id	title	n_citation	page_start	page_end	doc_type	publisher	volume	issue	doi	references	abstract
101421652	The influence of ...	139.0	397	423	Journal	Society for Infor...	27	3	10.2307/30036539	[1516261653, 1978...	Managers in moder...
1015675232	Research-paper re...	106.0	305	338	Journal	Springer Berlin H...	17	4	10.1007/s00799-01...	[1971040550, 2012...	In the last 16 ye...
10311529	Technical Section...	236.0	85	103	Journal	Pergamon Press, Inc.	33	1	10.1016/j.cag.200...	[2075597533, 2118...	User interfaces i...
104754383	Shackled to the s...	215.0	21	42	Journal	Society for Infor...	36	1	10.2307/41410404	[1758167268, 1918...	Given that adopti...
108157922	The State of the ...	114.0	141	178	Journal	Kluwer Academic P...	7	3	10.1023/A:1008287...	null	This paper develo...
112035675	Understanding fit...	414.0	167	193	Journal	Society for Infor...	25	2	10.2307/3250928	[1595507819]	Many previous pap...
115183765	Technostress: tec...	287.0	831	858	Journal	Society for Infor...	35	4	10.2307/41409963	[1510394286, 1721...	With the prolifer...
117893956	Survey paper: A s...	273.0	510	522	Journal	Elsevier Science ...	56	6	10.1016/j.compind...	[2041748119, 2171...	The research lite...
1219527654	Business intellig...	154.0	1189	1216	Journal	Society for Infor...	36	4	10.2307/41703504	[1565251461, 2033...	The increasing po...
12532419	Variational Optic...	215.0	245	255	Journal	Kluwer Academic P...	14	3	10.1023/A:1011286...	[1538585633, 1572...	Nonquadratic vari...
1276866904	Positioning and p...	665.0	337	356	Journal	M I S Research Ce...	37	2	10.25300/MISQ/201...	[1532730203, 1565...	Design science re...
127809959	Coherence-Enhanci...	594.0	111	127	Journal	Kluwer Academic P...	31	2	10.1023/A:1008009...	[1970269179]	The completion of...
134908229	The adoption and ...	154.0	289	323	Journal	Society for Infor...	27	2	10.2307/30036532	[112035675, 15955...	This paper report...
136147141	Ontology alignmen...	162.0	158	192	Journal	Springer, Berlin,...	15	6720	10.1007/978-3-642...	[2015191210, 2047...	In the area of se...
139533467	Password Authenti...	112.0	101	115	Journal	捷径科技有限公司	3	2	10.6633/IJNS.2006...	null	Password authenti...
141544107	Interactive decis...	115.0	293	320	Journal	Society for Infor...	33	2	10.2307/20650293	[1511558225, 1576...	This paper extend...
1422679120	Interval-valued i...	101.0	183	191	Journal	Elsevier	248	1	10.1016/j.ejor.20...	[2002574279, 2009...	Inter-dependency ...
1424318399	A review of defin...	125.0	47	61	Journal	Elsevier	145	145	10.1016/j.ress.20...	[1980156780]	Modeling and eval...
145670326	Survey paper: A s...	161.0	940	965	Journal	Elsevier North-Ho...	56	2	10.1016/j.comnet....	[2149863032, 2150...	Wireless networki...
1480498941	Automatic Detecti...	154.0	63	77	Journal	Kluwer Academic P...	32	1	10.1023/A:1008145...	[2134864374]	This paper demons...

2. **Drop the columns page_start and page_end from the paper data frame.**:

```
paper_df.drop("page_start","page_end") \
    .show()
```

Execution time: 181 ms ± 38.6 ms

3. **Insert an author in the author data frame.**:

```
newRow = spark.createDataFrame([(1882405, "Elon Musk", "Tesla")],
    ["id","name","org"])

new_df = author_df.union(newRow)
new_df.filter(new_df.name == "Elon Musk").show(truncate = False)
```

Execution time: 436 ms ± 29.2 ms

4. **Update the title of a paper with a particular id.**:

```
new_value = "New_Title"
new_paper_df = paper_df.withColumn(
    "title",
    when(
        col("id") == 101421652,
        new_value
    )
)

new_paper_df.filter(new_paper_df.id == 101421652).show()
```

Execution time: 250 ms ± 30.7 ms

5. **Update the title of all the papers with more than 100 pages.**:

```
new_value = "New_Title"
new_paper_df = paper_df.withColumn(
    "title",
    when(
        col("page_end") - col("page_start") > 100,
```

```
6           new_value
7       )
8   )
9
10  new_paper_df.filter(new_paper_df.title == new_value).show()
```

Execution time: 188 ms ± 24.7 ms

4 | References

- **AMiner data set V11**
- **Bio data set**
- **Tweets data set**
- **Images data set**
- **Draw.io**
- **Neo4j**
- **MongoDB**
- **Apache Spark**
- **Overleaf**
- **Python**
- **PyCharm**
- **Google Colaboratory**