POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Systems and Methods for Big and Unstructured Data Project

Author(s): **Arcaro Stefano**

Benzoni Lorenzo

Cartechini Giacomo

D'Onofrio Gianmarco

Dell'Agosto Giacomo

Group Number: **1**

Academic Year: 2022-2023

Contents

1 | Introduction

1.1. Problem description

We have a large dataset containing information about scientific publications, their authors and publishers, as well as information about books written by those authors, proceedings, journals and thesis.
We are required to build a system where adding new data, uploading and querying existing data is as straightforward as possible. An example of a well-built system is certainly the DBLP database, which is the database we referenced to build the backbone of our solution.

1.2. Assumptions

[School, Journal, Publisher, Series, Author, Editor] have multiplicity 1:N in their relationship with Divulgation, as we believe it doesn't make sense for a database entry to not be connected to any other entity. The correlation between Journal and Publisher is checked before database upload; they are then logically connected in the diagram through the Publication entity. A Thesis (be it PhD or Master's) can only be submitted at one school, where it was written.

2 | Entity-Relationship Model

2.1. Description

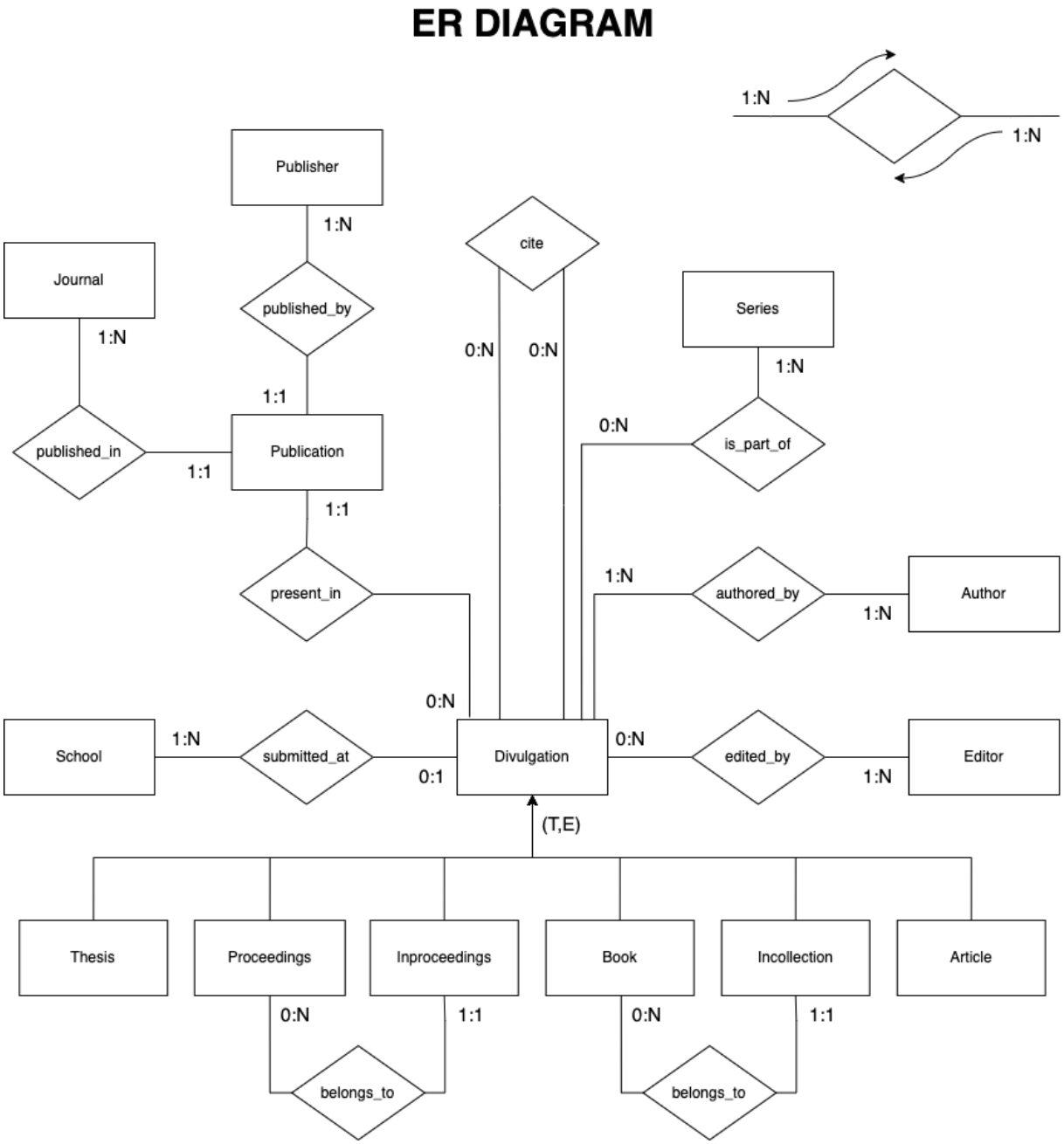
The process of creating our diagram began with the identification of the main entities revolving around what we knew about a bibliography. We, therefore, started modelling entities such as authors and editors, as well as scientific journals with their respective publishers. All of these are logically connected to a general Divulgation entity representing any kind of scientific work. This rough sketch was then expanded and elaborated upon, with some research and looking at the DBLP database, by adding universities and series and all the divulgation types, such as thesis, books and articles. We then finalised the diagram by adding a primary key and all the required attributes to the modelled entities.

2.2. Notes

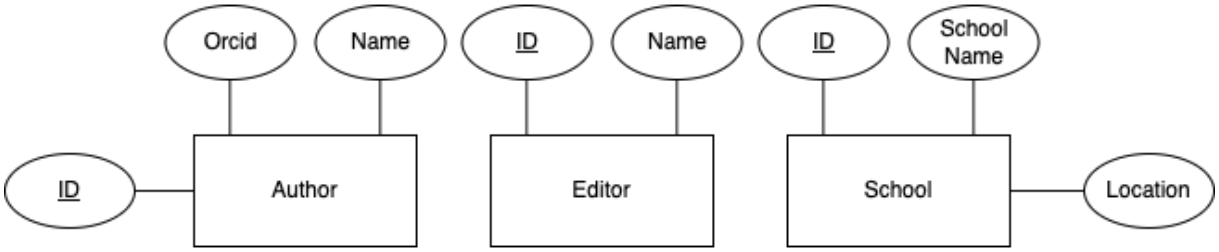
- While the DBLP database contains a Cite entity with an array of keys referring to the cited records (Divulgation entities), we believe a more "ER-oriented" approach is to have a Divulgation directly cite another entity of the same type with an N:N multiplicity
- The Divulgation entity only has one mandatory relationship, the one with the Author(s); all the other relationships have a 0:N multiplicity, given the fact that they are not shared by all the different kinds of Divulgation
- For the Divulgation entity, we chose the ee (electronic edition) attribute to be its primary key: the DOI was initially chosen, but seeing as not every document has an assigned DOI, we chose to go with the more general link to the document's resource
- A similar situation to the one above happened with the Author entity: while the initially chosen primary key was the Orcid, a unique identifier for every researcher/author of scientific works, we decided to use a more generic incremental integer ID: this choice was made for the database to be compatible with decades-old publications, whose authors did not have an Orcid and might have never gotten one

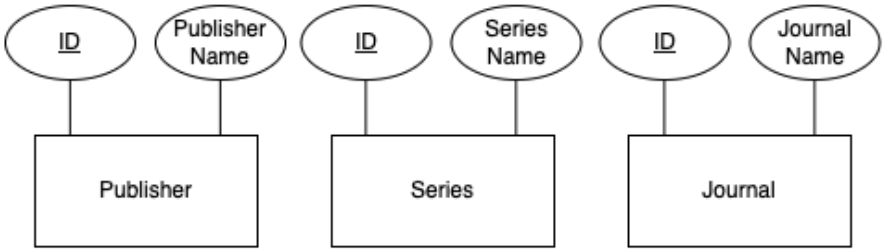
- Both the "belongs_to" relationships are to be read from Inproceedings (Incollection) to Proceeding (Book); the reason for choosing the 0:N multiplicity on the side of the Proceedings (Book) instead of 1:N is that these two could exist even without being a collection of titled sections
- An incremental ID was chosen for the [Editor, School, Publisher, Series, Journal] entities as well since their only meaningful attributes for the sake of this database might not be unique (such as a name)
- The url attribute of the Divulgation entity describes the location of the document in the DBLP website, where we got the database entries from
- The crossref attribute of Inproceedings (Incollection) refers to the Proceedings (Book) entity containing it

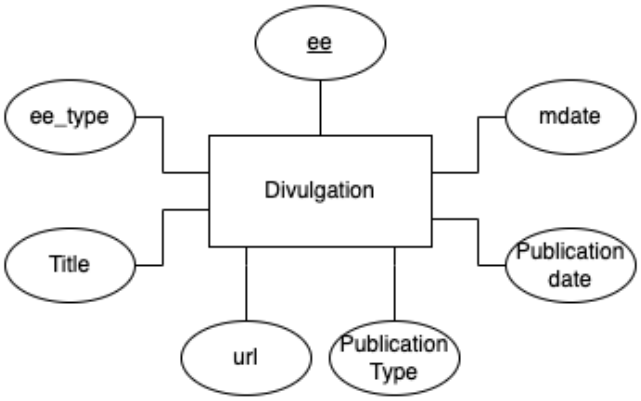
2.3. Diagram representation

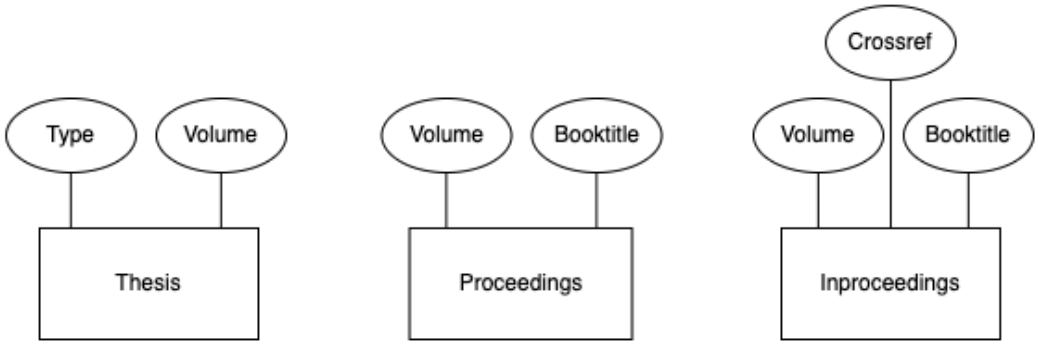


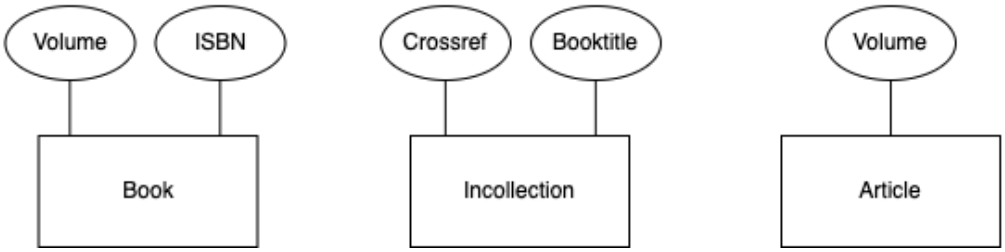
ATTRIBUTES


Orcid
Name
ID
Author
ID
Name
Editor
ID
School Name
School
ID
Location


ID
Publisher Name
Publisher
ID
Series Name
Series
ID
Journal Name
Journal


ee
ee_type
mdate
Divulgation
Title
Publication date
url
Publication Type


Type
Volume
Thesis
Volume
Booktitle
Proceedings
Crossref
Volume
Booktitle
Inproceedings


Volume
ISBN
Book
Crossref
Booktitle
Incollection
Volume
Article

3 | Neo4J

3.1. Data Ingestion

The source of our data is the DBLP database. The data can be downloaded from the website in a XML format and then converted into CSV format with this tool. Then the command used to populate the database is:

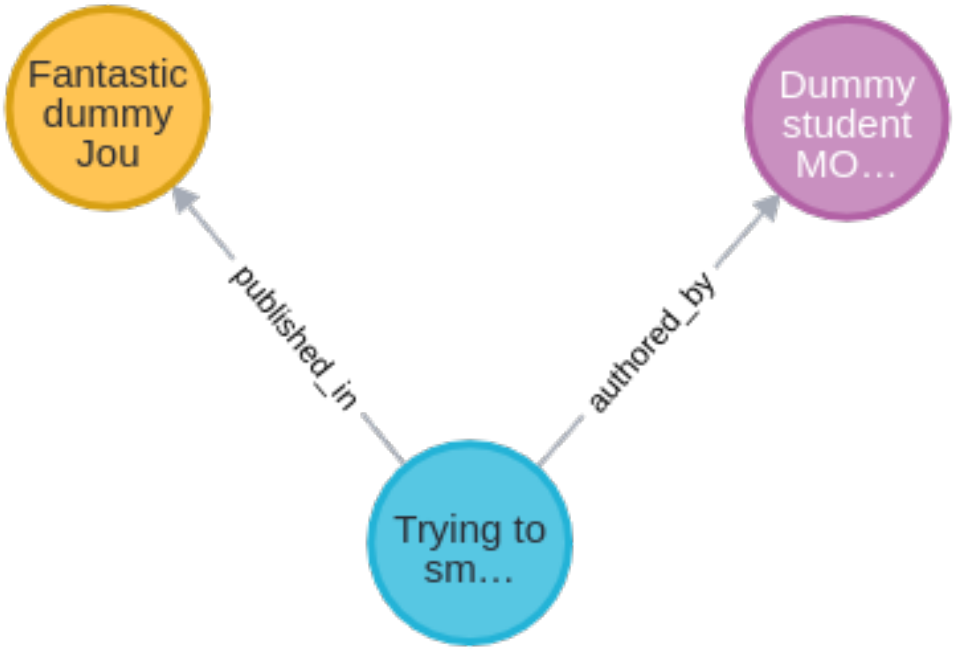
```
$ neo4j-admin import
--database=neo4j.db
--delimiter ";"
--array-delimiter "|"
--id-type INTEGER
--nodes=mastersthesis=
    "output_mastersthesis_header.csv,output_mastersthesis.csv"
--nodes=article="output_article_header.csv,output_article.csv"
--nodes=incollection=
    "output_incollection_header.csv,output_incollection.csv"
--nodes=phdthesis="output_phdthesis_header.csv,output_phdthesis.csv"
--nodes=book="output_book_header.csv,output_book.csv"
--nodes=inproceedings=
    "output_inproceedings_header.csv,output_inproceedings.csv"
--nodes=www="output_www_header.csv,output_www.csv"
--nodes=proceedings=
    "output_proceedings_header.csv,output_proceedings.csv"
--nodes=school="output_school.csv"
--relationships=submitted_at="output_school_submitted_at.csv"
--nodes=editor="output_editor.csv"
--relationships=edited_by="output_editor_edited_by.csv"
--nodes=series="output_series.csv"
--relationships=is_part_of="output_series_is_part_of.csv"
--nodes=publisher="output_publisher.csv"
--relationships=published_by="output_publisher_published_by.csv"
```

```
--nodes=author="output_author.csv"
--relationships=authored_by="output_author_authored_by.csv"
--nodes=cite="output_cite.csv"
--relationships=has_citation="output_cite_has_citation.csv"
--nodes=journal="output_journal.csv"
--relationships=published_in="output_journal_published_in.csv"
```

3.2. CREATE queries

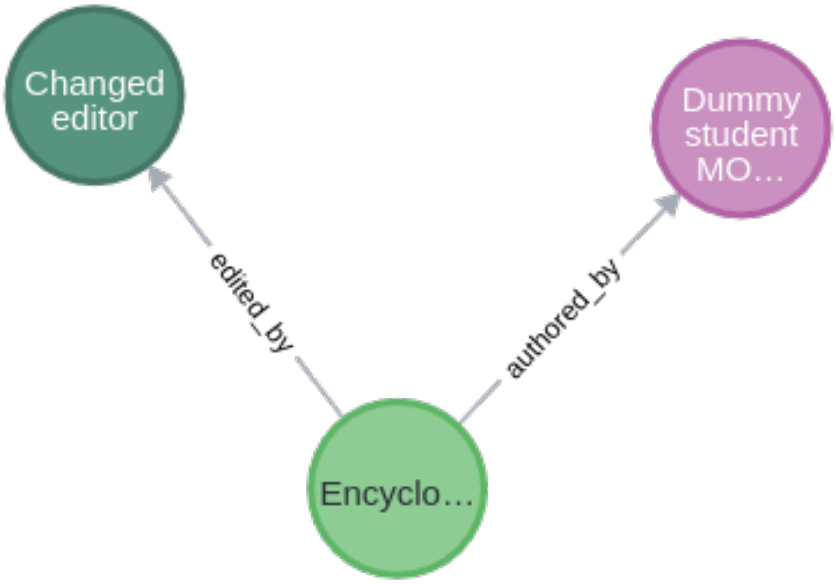
3.2.1. Simple create query

```
CREATE (a: article
       {
           article: 123123123123,
           author: ["Mock Author"],
           journal :"GTE Laboratories Incorporated",
           mdate :"2022-10-25",
           month :"October",
           publtype :"informal",
           title :"Trying to create smt",
           url   :["db/journals/gtelab/index.html#TM-0014-06-88-165"],
           volume  :"TM-0014-06-88-165",
           year  :2022
       }
    ) - [:authored_by] -> (au: author
    {
        author: "Dummy student MOCK"
    }
)

CREATE (a) - [:published_in] -> (j: journal {
    journal: "Fantastic dummy Journal"
})

RETURN a, au, j
```



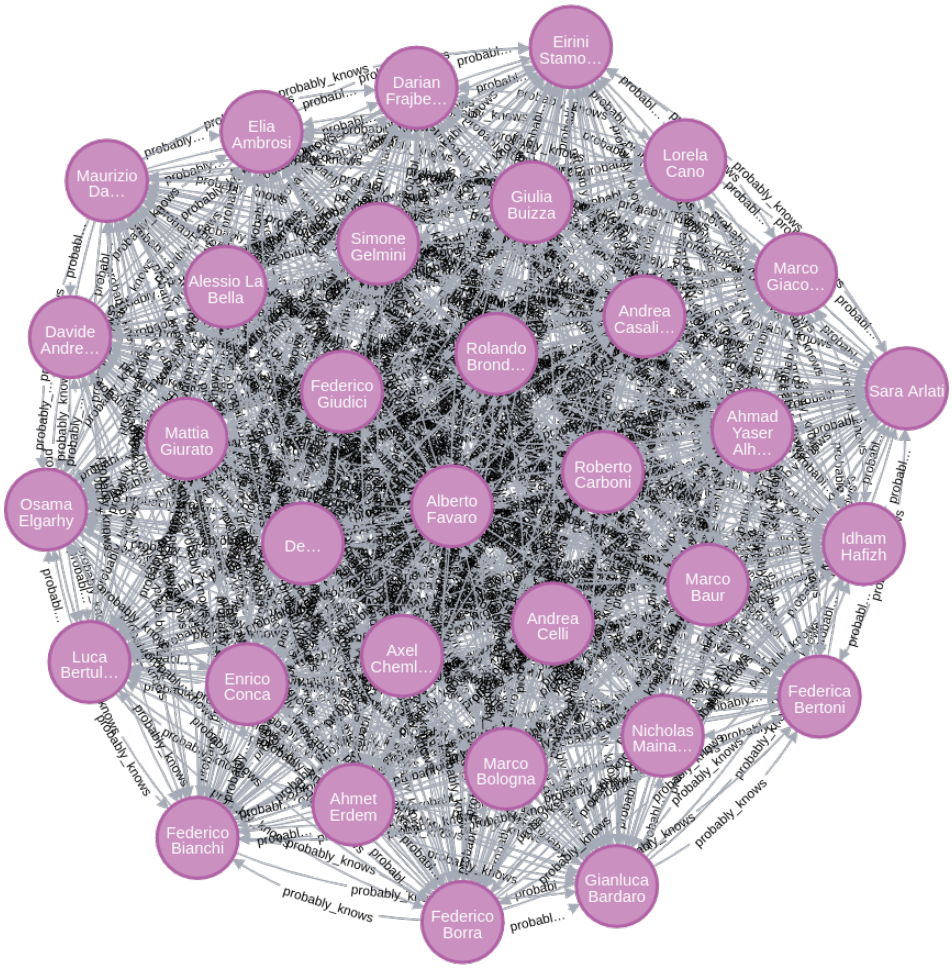
3.2.2. Add Author to an already inserted book, and change its author

```
MATCH (b:book {booktitle: "Encyclopedia of Information Ethics and Security"})

MATCH (b) - [:edited_by] -> (e: editor)
SET e.editor = "Changed editor"

MATCH (b) - [:authored_by] -> (au: author) {
    author: "Dummy student MOCK"
})

RETURN b, au, e
```



3.2.3. Create new relation between authors

```
MATCH (s:school) <- [:submitted_at] - (phdt: phdthesis) - [:authored_by] ->
    (author: author)
WHERE s.school =~ '.*Milan.*' AND 2020 IN phdt.year
WITH collect (author) AS milanAuthors
foreach (au1 IN milanAuthors |
    foreach(au2 IN milanAuthors |
        MERGE (au1) - [:probably_knows] -> (au2)
    )
)

// Removes reflective property
WITH milanAuthors AS MA
MATCH (au: author) - [r: probably_knows] -> (au: author)
DELETE r

RETURN MA
```



3.2.4. Create a Series, a Publisher, a Book, an Editor and an Author

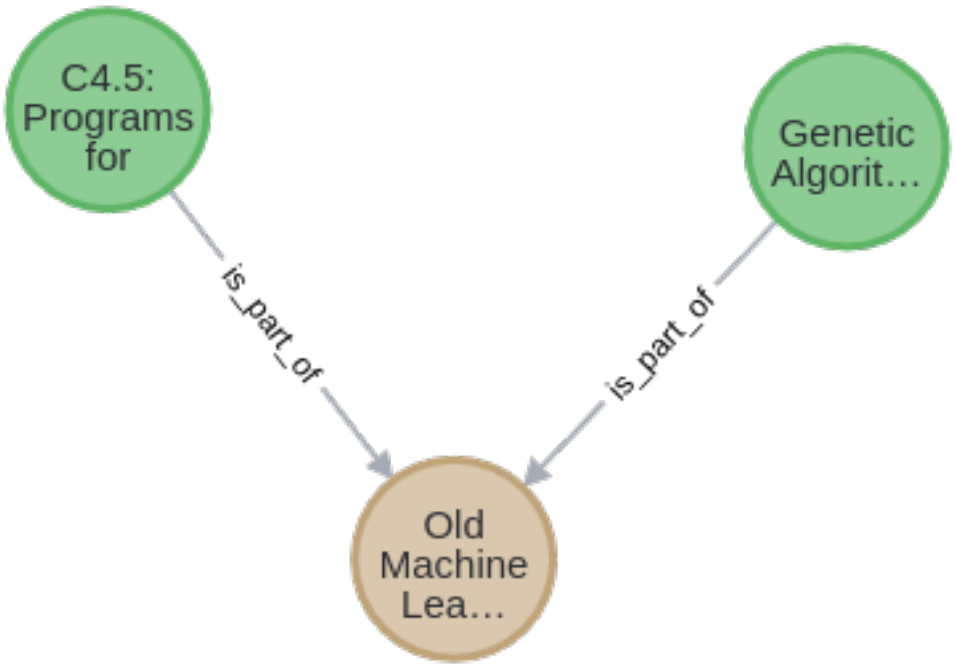
```
CREATE (p: publisher {
    publisher: 'Polimi publisher'
})

CREATE (b: book {
    title: 'Amazing book'
})

CREATE (s: series {
    series: 'An Amazing series'
})

CREATE (s) <- [:is_part_of] - (b) - [:published_by] -> (p)
```

```
CREATE (b) - [:edited_by] -> (e: editor { editor: 'Polimi editor'})
CREATE (b) - [:authored_by] -> (a: author { author: 'Amazing author' })

SET b.isbn = '978-1-4200-3531-5,978-1-58488-301-2'

RETURN *
```

3.2.5. Create a Machine Learning series, and attach all the books that contain the keyword 'Machine Learning' in the book title

Part 1

```
CREATE (os: series { series: 'Old Machine Learning'})
WITH os AS OldMachineLearningSeries

MATCH (book: book)
WHERE book.title =~ '.*(?i)Machine Learning' AND book.year < 2000
CREATE (book) - [:is_part_of] -> (OldMachineLearningSeries)

RETURN *
```



Part 2

```
CREATE (s: series { series: 'Machine Learning'})
WITH s AS MachineLearningSeries

MATCH (book: book)
WHERE book.title =~ '.*(?i)Machine Learning' AND book.year > 2000

CREATE (book) - [:is_part_of] -> (MachineLearningSeries)

RETURN *
```

3.3. UPDATE queries

Thanks to the APOC plugin we've been able to enrich our database. Our goal was to add some geographical information about the schools of our dataset, this was achived by the geocoding API of GMaps.

We have performed these update queries to create a more specific geospatial representation of the data. Most of the queries required adequate geospatial precision, which was not present in the original data.

3.3.1. Add latitude/longitude

```
CALL apoc.periodic.iterate('MATCH (a:school) RETURN a',
  'CALL apoc.spatial.geocodeOnce(a.school) YIELD location
  WITH a, location.latitude AS latitude, location.longitude AS longitude
  SET a.latitude = latitude,
    a.longitude = longitude', {batchSize:100, parallel:true})
```

3.3.2. Add country

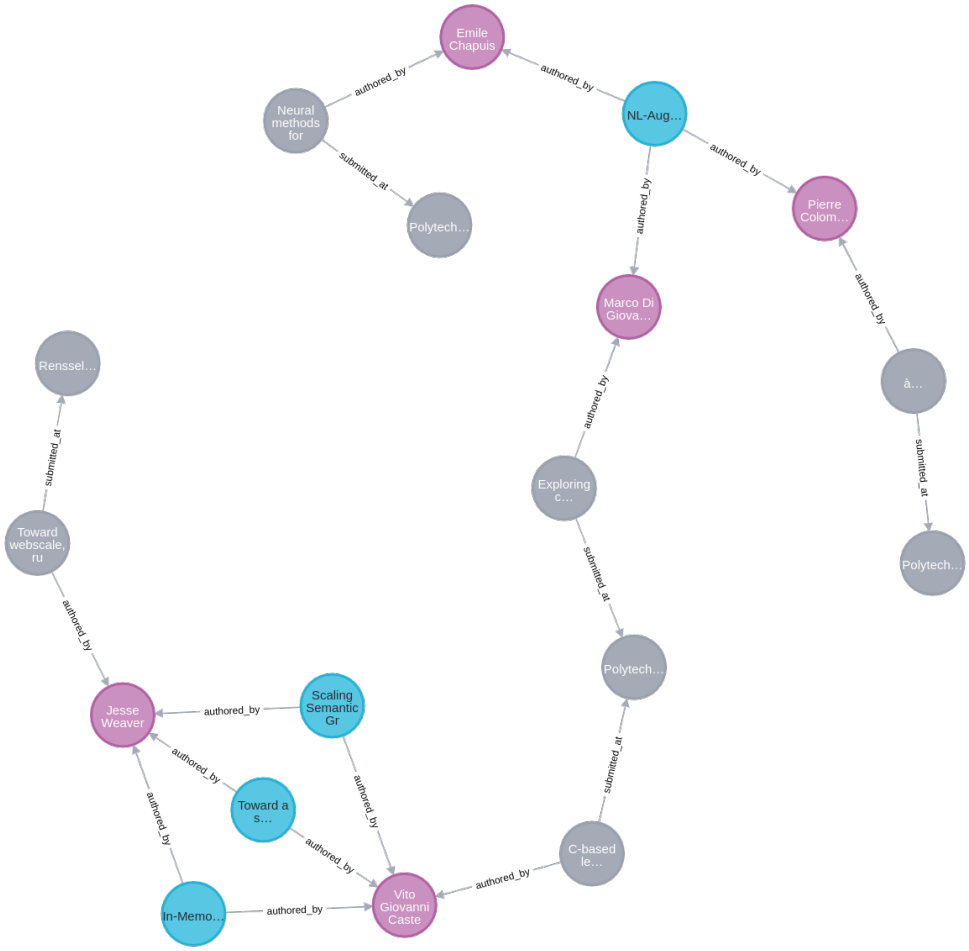
```
CALL apoc.periodic.iterate('MATCH (a:school) RETURN a',
  'CALL apoc.spatial.geocodeOnce(a.school) YIELD data
  UNWIND data.address_components as x
  WITH x, a
  WHERE x.types[0] = "country"
  SET a.country = x.long_name', {batchSize:100, parallel:true})
```

3.4. Queries

3.4.1. Poly-bros

Authors who submitted a PhD thesis at PoliMi and then also co-wrote articles with authors who submitted a PhD thesis at other Polytechnical universities around the world

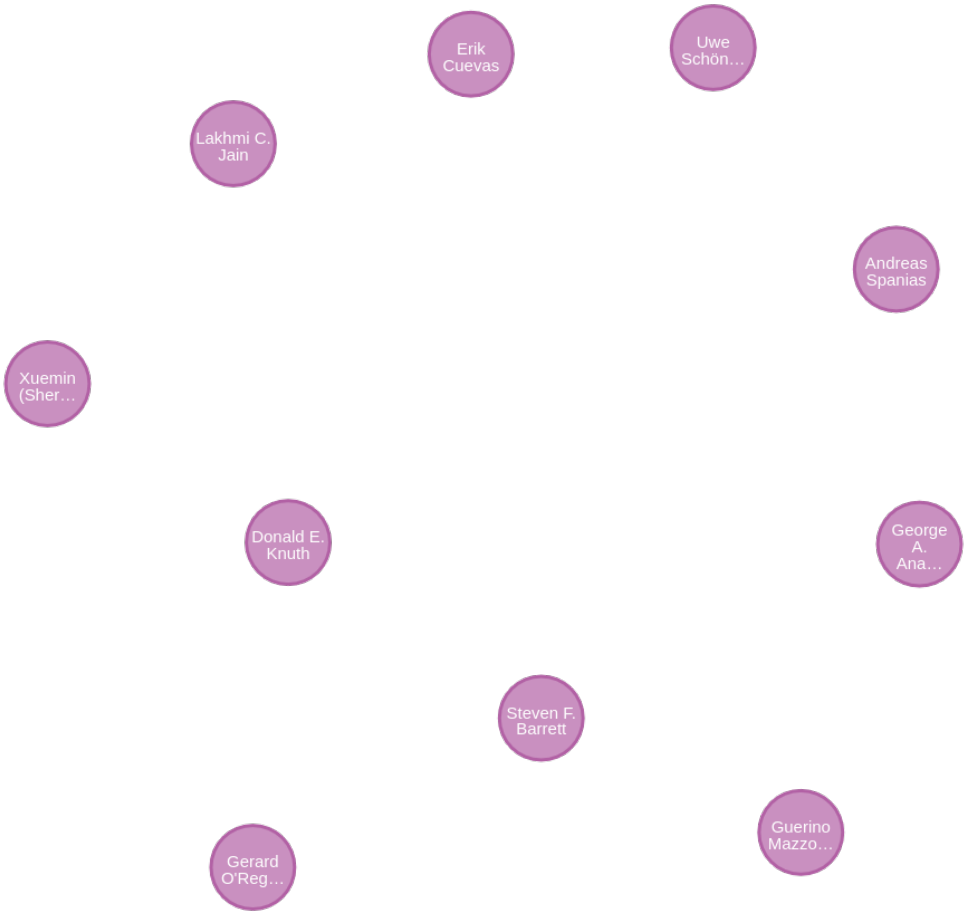
```
MATCH
    p = (poly:school)<-[:submitted_at]-(:phdthesis)-[:authored_by]->(:author)<-
        [:authored_by]-(:article)-[:authored_by]->(poly_author:author)<-
        [:authored_by]-(:phdthesis)-[:submitted_at]->(polimi:school)
WHERE
    poly.school =~ '.*Polytechnic.*' AND
    NOT (poly.school =~ '.*Polytechnic.*Milan.*') AND
    polimi.school =~ '.*Polytechnic.*Milan.*'
RETURN poly_author, p
```



3.4.2. Serial writers

Top 10 authors that have written the most books belonging to a series

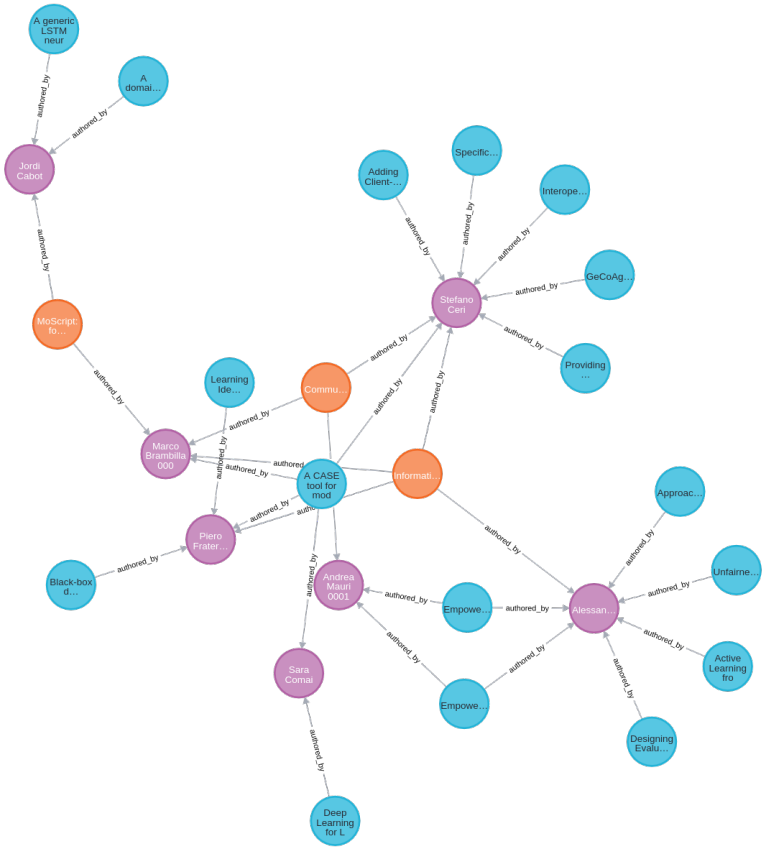
```
MATCH p = (a:author)<-[:authored_by]-(b:book)-[:is_part_of]->(s:series)
WITH a, count(b) AS n_books
ORDER BY n_books DESCENDING
RETURN a, n_books
LIMIT 10
```



3.4.3. AI enthusiasts

AI-related articles written by authors related to Marco Brambilla

```
MATCH p = (pt:article)-[:authored_by]->(a:author)-[r*1..3]-(mb:author)
WHERE
    (
        NONE (rel IN r WHERE type(rel)="published_in") AND
        mb.author = 'Marco Brambilla 0001'
    ) AND
    (
        pt.title =~ '.* (?i)A(?i)I .*' OR
        pt.title =~ '.*(?i)Artificial (?i)Intelligence.*' OR
        pt.title =~ '.*(?i)Learning.*' OR
        pt.title =~ '.*(?i)Agent.*' OR
        pt.title =~ '.*(?i)Neural.*'
    )
RETURN pt, p
LIMIT 1000
```



3.4.4. Touching the stars

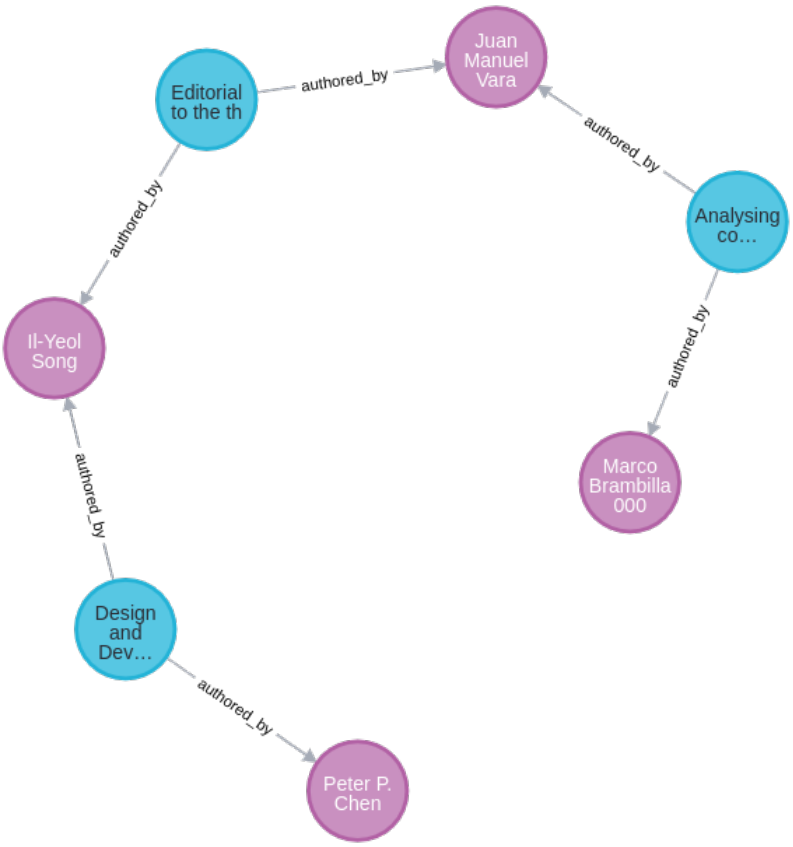
Find a strong connection (made of article and authors) between the author of the most cited article and Marco Brambilla.

The first query counts for every citation that is not "..." the number of incoming relations of type "has_citation" and saves the value into countRel. Then all citations are ordered by decreasing countRel and only the first one is kept.

The second matched pattern finds all articles and their authors where the key of the article corresponds to the field "cite" of the most cited citation previously found and the author's name is saved in auth.

Finally, starting from the author node with value "Marco Brambilla 0001", the last query find the shortest path to auth passing only through nodes that are either of type "author" or "article" and returns the whole path.

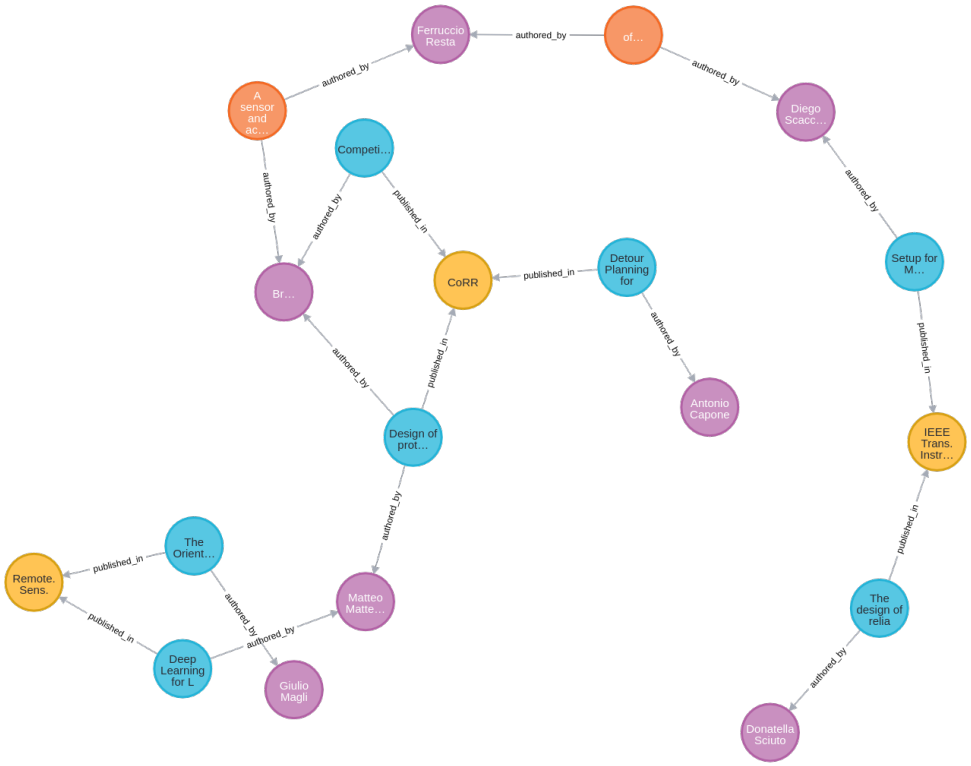
```
MATCH (cit:cite)<-[r:has_citation]-(any)
WHERE cit.cite <> '...'
WITH cit, COUNT(r) AS countRel
ORDER BY countRel DESC
LIMIT 1

MATCH (n:article)-[:authored_by]->(auth:author)
WHERE n.`key` = cit.cite
WITH auth

MATCH p = shortestPath((rest:author {author: 'Marco Brambilla 0001'})
    -[*]-(auth))
WHERE ALL (x IN nodes(p) WHERE x:author OR x:article)
RETURN p
```



3.4.5. Major players

Shortest path between the actual rector and all the candidates

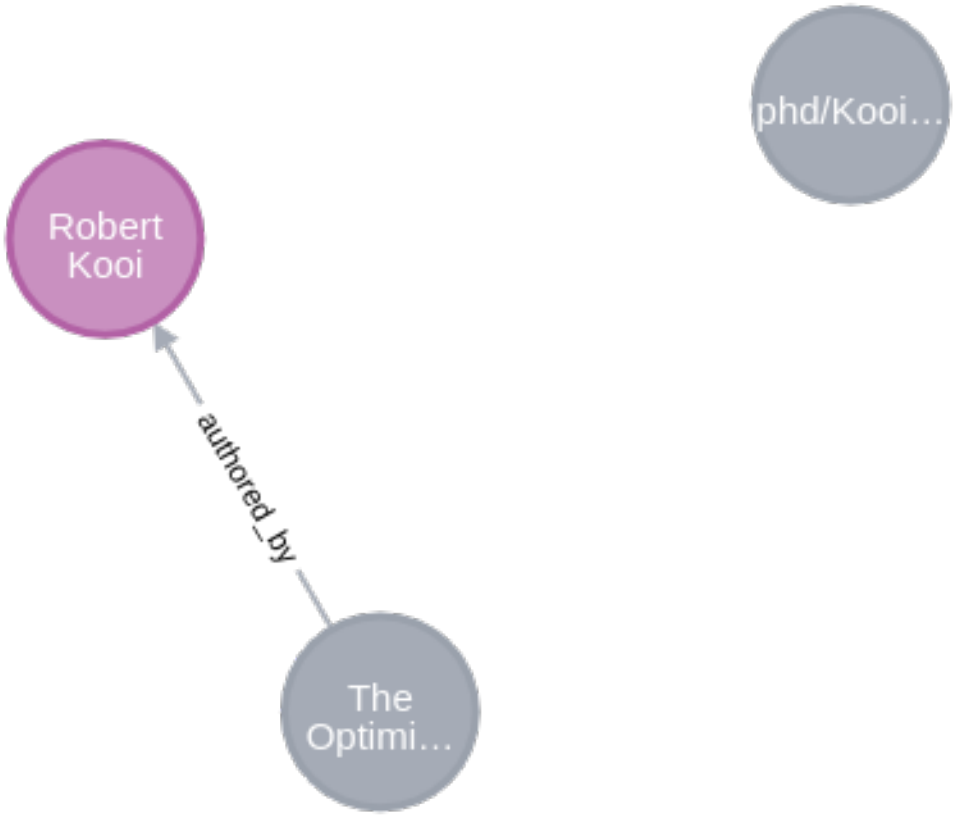
```
MATCH (candidate:author)
WITH ['Antonio Capone', 'Donatella Sciuto', 'Giulio Magli'] AS candidates,
    candidate
WHERE candidate.author IN candidates
WITH candidate

MATCH p = shortestPath((candidate) - [*] - (rector:author {author: 'Ferruccio
    Resta'}))
RETURN candidate, size(nodes(p)) AS distance, p
ORDER BY distance
```



3.4.6. PhD dragon

The author of the thesis that has been cited the most

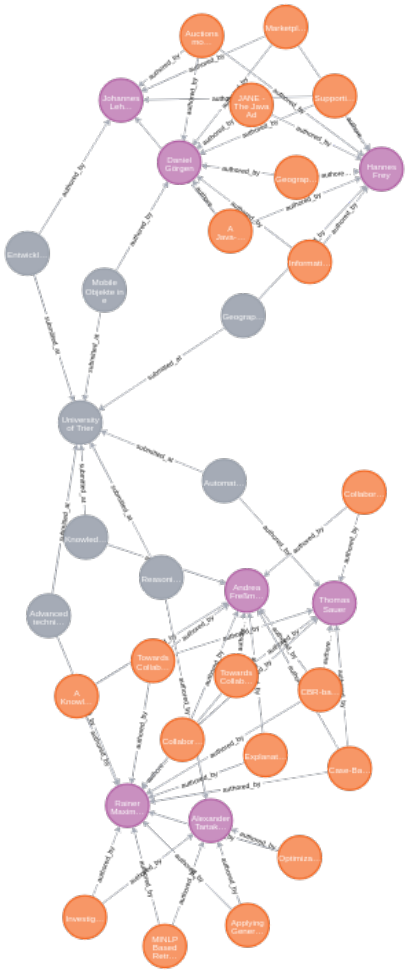
```
MATCH (sub)-[:submitted_at]->(sch:school)
WITH collect(sub.`key`) AS keyFromSc

MATCH (cit:cite)<-[r:has_citation]-(any)
WHERE cit.cite IN keyFromSc
WITH count(r) AS c, cit
ORDER BY c desc
LIMIT 1

MATCH (n)-[:authored_by]->(auth:author)
WHERE n.`key` = cit.cite
RETURN n, cit, auth, c
```



3.4.7. UniBros

The path between pairs of authors who are affiliated with the same university and have co-authored an inproceedings

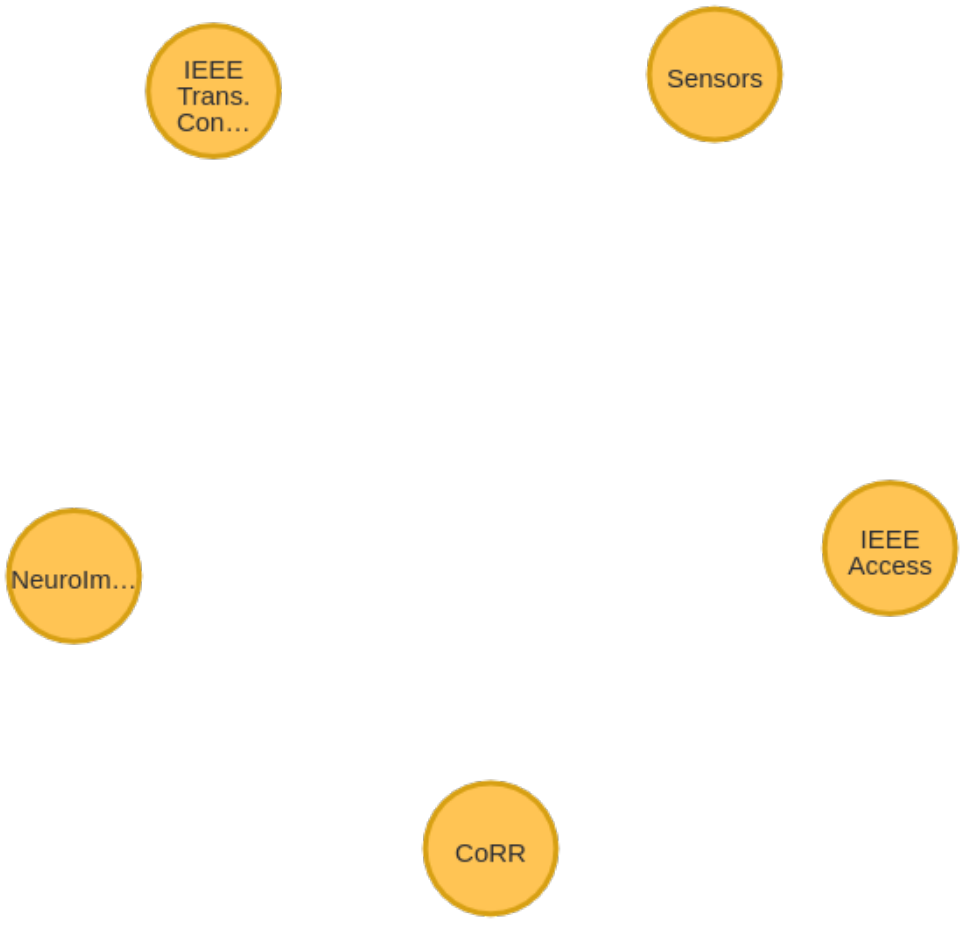
```
MATCH p = (pr:inproceedings)-[:authored_by]->(a1:author)<-[:authored_by]-
    (thesis1)-[:submitted_at]->(sch:school)<-[:submitted_at]-(thesis2)-
    [:authored_by]->(a2:author)<-[:authored_by]-(pr:inproceedings)
RETURN p
LIMIT 50
```


Auctions mo...
Marketpl...
Johannes Leh...
JANE – The Java Ad
Support...
Daniel Görgen
Geograp...
Hannes Frey
A Java-...
Informati...
Entwicklu...
Mobile Objekte in e
Geograp...
University of Trier
Automat...
Collabor...
Knowled...
Ressourc...
Andrea Fradin...
Thomas Sauer
Advanced techni...
Towards Collab...
Towards Collab...
CSR-ba...
A Knowl...
Collabor...
Explanat...
Case-Ba...
Rainer Maxim...
Alexander Tartak...
Optimiza...
Investig...
Applying Conver...
MINLP Based Restr...

3.4.8. PoliMi dominance

The journals with the most articles written by people affiliated with PoliMi

```
MATCH (j:journal)<-[:published_in]-(art:article)-[:authored_by]->
(auth:author)<-[:authored_by]-()-[:submitted_at]->(s:school)
WHERE s. school =~ '.*Polytechnic.*Milan.*'
WITH count (distinct (art)) AS narticles, j
ORDER BY narticles desc
RETURN j, narticles
LIMIT 5
```



3.4.9. GG Authors

Authors who wrote at least 10 books (total) that are part of series in the top 10 for the most books.
The first query in the pipeline finds all series and their books, counts the number of books in each series and assigns it to nbooks, then orders the series by decreasing nbooks and saves only the top 10 in the list topSeries. The second query matches a pattern that finds the authors of books part of series in topSeries and for each author counts the books in the pattern and assigns it to nbooks. Finally, if nbooks is greater than 10, the author and nbooks are returned.

```
MATCH (s:series)<-[:is_part_of]-(b:book)
WITH count(b) AS nbooks, s
ORDER BY nbooks desc
LIMIT 10
WITH collect(s) AS topSeries

MATCH (ts:series)<-[:is_part_of]-(b:book)-[:authored_by]->(author:author)
WHERE ts IN topSeries
WITH COUNT(b) AS nbooks, author
WHERE nbooks >= 10
RETURN author, nbooks
```

"author"	nbooks
"author":"George A. Anastassiou"	13
"author":"Erik Cuevas"	11

Table 3.1: Returned values

3.4.10. AI enthusiasts 2

Number of AI-related articles written by at least two authors each year

```
MATCH (auth1)<-[:authored_by]-(pt:article)-[:authored_by]->(auth2:author)
WHERE
    pt.title =~ '.* (?i)A(?i)I .*' OR
    pt.title =~ '.*(?i)Artificial (?i)Intelligence.*' OR
    pt.title =~ '.*(?i)Learning.*' OR
    pt.title =~ '.*(?i)Agent.*' OR
    pt.title =~ '.*(?i)Neural.*'
```

```
RETURN DISTINCT(pt.year), COUNT(DISTINCT(pt))
ORDER BY pt.year DESC
```

3.4.11. PhD neighbours

Find authors related to another author that have submiteed a PhD Thesis at two different italian universities in a 10 km radius

```
match(s:author)<-[:authored_by]-(phd:phdthesis)-[:submitted_at]->(sch:school
    {country: 'Italy'})
with s, point({latitude: sch.latitude, longitude: sch.longitude}) as uni, sch
match(s2:author)<-[:authored_by]-(phd2:phdthesis)-[:submitted_at]->(sch2:school
    {country: 'Italy'})
with point({latitude: sch2.latitude, longitude: sch2.longitude}) as nearUni,
    uni, s, sch, s2
where point.distance(uni, nearUni) < 10000 and sch <> sch2
return distinct(s.author) as Author, 'has done phd near', collect(s2.author)
    as Authors
```

"Author"	"'has done phd near'"	"Authors"
"Wolfgang Weber 0003"	"has done phd near"	["Jolanda Patruno","Andrea Cherubini","Luna Dimitrio","Giovanni Farina","Emanuele Natale","Nicolo Rivetti","Antonio Piccolomini d'Aragona","Nicole Dore","Mario Lezoche","Andrei Dorman"]
"Luigi Liquori"	"has done phd near"	["Emanuele Ruffaldi","Razvan Andrei Popescu","Antonio Gulli","Giulio Ermanno Pibiri"]
"Angelo Troina"	"has done phd near"	["Emanuele Ruffaldi","Razvan Andrei Popescu","Antonio Gulli","Giulio Ermanno Pibiri"]
"Jesús González-Feliu"	"has done phd near"	["Marco Naddeo"]
"Ihab Makki"	"has done phd near"	["Marco Naddeo"]
"Marco Cisternino"	"has done phd near"	["Marco Naddeo"]

4 | Data Visualization

We have created and HTML + JS view that fetches data from the neo4j database. This opens endless possibilities in the field of data visualization. In the next deliveries we will dive deeper into this topic. More on our github repo.

4.1. Heatmaps

We created a Map, with a heatMap layer showing the world's most influential universities based on total number of PhD publications.

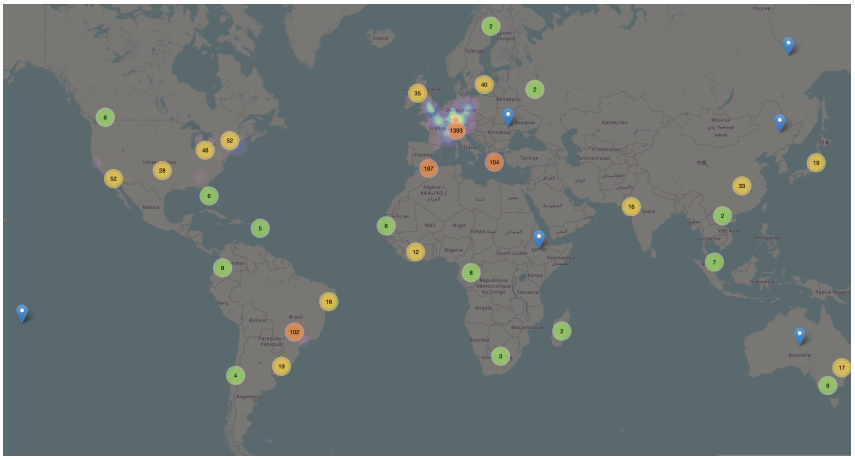


Figure 4.1: Markers are universities.

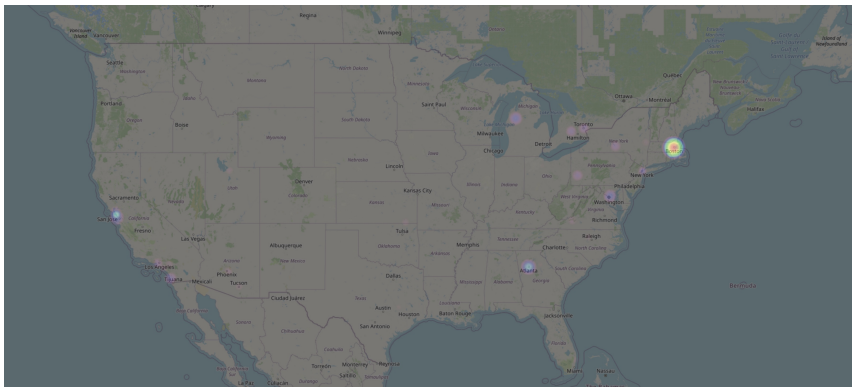


Figure 4.2: United States of America

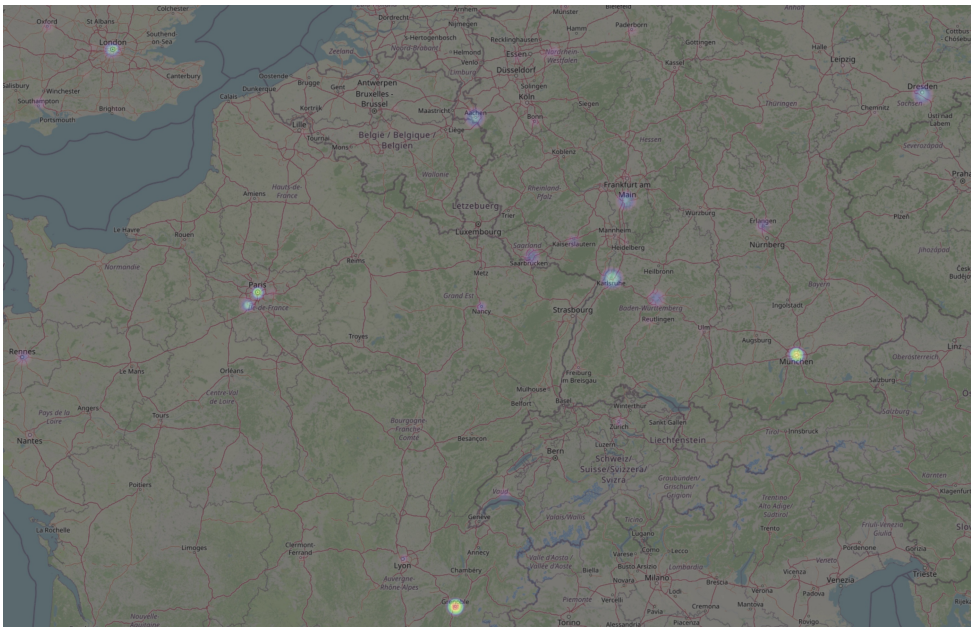


Figure 4.3: North Europe

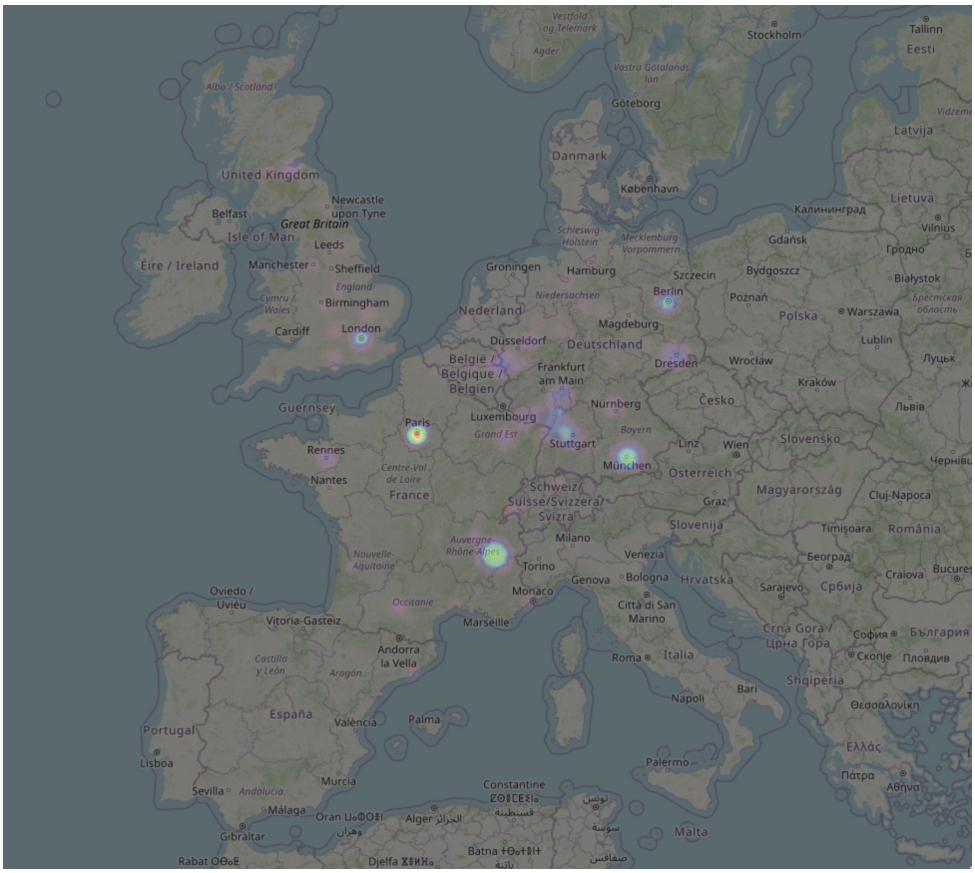


Figure 4.4: Europe