UNIVERSITÀ DEGLI STUDI DI NAPOLI

FEDERICO II

Scuola Politecnica e delle Scienze di Base

Corso di Laurea Magistrale in Ingegneria Informatica

Homework 2 – Big Data Engineering

Anno Accademico 2022-2023

Riccio Emanuele – M63001339

Tammaro Ferdinando – M63001380

Prof. Moscato Vincenzo

Sommario

Introduzione 3
Business Understanding 3
MongoDB 4
Setup 4
Query 5
Neo4J 9
Setup 9
Query 11
Grafici 14

Introduzione

Nel secondo Homework del Corso di Big Data Engineering, bisognava effettuare uno studio sui principali ambiti dei progetti intrapresi dall'Università Federico II di Napoli, partendo dal dataset fornitoci per il precedente Homework.

Per effettuare ciò sono stati utilizzati i database *MongoDB*, un database non relazionale, e *Neo4J,* un database a grafo.

Business Understanding

Per lo scopo prefissato abbiamo deciso di effettuare delle operazioni di pre processing sui dati; sono quindi state eliminate diverse colonne da noi ritenute superflue ai fini della nostra elaborazione.

Le colonne da noi conservate sono state:

- Grant ID: codice univoco identificativo
- Funding amount
- Start Year
- End Year
- Research Organization: i nomi delle organizzazioni coinvolte nei progetti, separate da un punto e virgola
- Funder
- Fields of Reaserch: tutti i campi di studio inerenti al progetto, separati da un punto e virgola

MongoDB

Il database è stato installato in cloud sfruttando il piano Atlas di MongoDB. L'accesso è stato gestito tramite la libreria PyMongo utilizzando il linguaggio Python.

Setup

```
1. from pymongo.mongo_client import MongoClient
2. import pymongo
3.
4. uri =
   "mongodb+srv://eleunameoiccir:<password>@cluster0.56jlhub.mongodb.net/?retryWr
   ites=true&w=majority"
5. client = MongoClient(uri)
```

Da Atlas MongoDB abbiamo ottenuto la stringa di connessione, tramite la quale è stato possibile connettersi al database cloud.

```
1. def init_db():
2.     #se non esistono db e collezione sono create
3.     db = client.unina
4.     docu = db.docu
5.     s = "GrantID;Funding Amount in EUR;Start Year;End Year;Research
   Organization;Funder;Fields of Research";
6.     header = list(s.split(";"))
7.     print(header)
8.     csvFile = open('data_mongo.CSV', 'r')
9.     reader = csv.DictReader(csvFile, delimiter=';')
10.    toSplit = [header[4], header[6]];
11.
12.    for each in reader:
13.        row = {}
14.        for field in header:
15.            row[field] = each[field]
16.        for f in toSplit:
17.            if not (row[f] and row[f].strip()):
18.                row[f] = list()
19.                continue
20.
21.            row[f] = list( map(lambda x: x.strip(), row[f].split(";")) )
22.
23.        docu.insert_one(row)
```

È stato creato e selezionato un database su MongoDB, per poi istanziare la collection *docu*.

Figura 1 - Esempio di Documento su MongoDB

L'import del CSV sul database è stato eseguito sempre tramite Python, andando a leggere riga per riga il file di input e convertendola in un dizionario Python, con una particolare attenzione per i campi che contenevano più dati separati da un punto e virgola, come Fields or Reaserch e Reaserch Organization, per le quali è stato effettuato uno split, trasformandole in liste. In questo modo, effettuare l'elaborazione di questi campi tramite query è risultata più semplice.

Query

Query 1

La prima query effettuata consiste nel contare le occorrenze di ogni *Field of Reaserch* tra tutti i progetti in cui era coinvolta l'Università Federico II. Per fare ciò, è stata realizzata una pipeline con diversi step.

Il primo di questi consisteva nel fare un filtraggio tra tutti quanti i progetti in cui risultava "University of Naples Federico II" tra le università partecipanti, tramite l'operatore `$match`. Ovviamente, essendo un dataset fornito dalla stessa Federico II, questa era presente in ogni progetto di ricerca, ma per fornire una soluzione più generale è stato scelto comunque di effettuare questo filtraggio.

In seguito è stato eseguita un'operazione di `$unwind`, che "distende" l'array `Fields of Reaserch` in modo che venga creata una replica del documento per ogni elemento presente nel campo su cui si è fatto l'unwind. Così facendo è possibile effettuare l'operazione di $group sul singolo argomento di ricerca, e contare il numero di documenti in ogni gruppo per assegnarlo alla variabile `field_count`. Il conteggio è effettuato tramite la funzione di accumulazione $sum.

Dopo aver ordinato in ordine decrescente tutti i dati con l'operazione `$sort`, viene eseguita la pipeline sui documenti della collezione tramite l'operazione `aggregate()`.

L'output è visibile in Figura 2.

```
1. def countTopics():
2.     db = client.unina
3.     docu = db.docu
4.     filter_fedeii = {
5.         "$match": {
6.             "Research Organization": {
7.                 "$in": ["University of Naples Federico II"]
8.             }
9.         }
10.    }
11.    un = {
12.        "$unwind": "$Fields of Research"
13.    }
14.    group_count = {
15.        "$group": {
16.            "_id": "$Fields of Research",
17.            # Count the number of movies in the group:
18.            "field_count": {"$sum": 1},
19.        }
20.    }
21.    sort = {
22.      "$sort": {
23.         "field_count": pymongo.DESCENDING
24.      }
25.    }
26.    pipeline = [
27.        filter_fedeii,
28.        un,
29.        group_count,
30.        sort,
31.    ]
32.    out = docu.aggregate(pipeline)
33.    for o in out:
34.        print("%s, %s" % (o["_id"], o["field_count"]))
```

```
31 Biological Sciences, 581
32 Biomedical and Clinical Sciences, 540
40 Engineering, 489
34 Chemical Sciences, 248
3101 Biochemistry and Cell Biology, 211
46 Information and Computing Sciences, 164
30 Agricultural, Veterinary and Food Sciences, 157
44 Human Society, 153
37 Earth Sciences, 135
33 Built Environment and Design, 129
3105 Genetics, 126
43 History, Heritage and Archaeology, 125
51 Physical Sciences, 117
41 Environmental Sciences, 112
49 Mathematical Sciences, 111
35 Commerce, Management, Tourism and Services, 109
3211 Oncology and Carcinogenesis, 88
48 Law and Legal Studies, 86
3202 Clinical Sciences, 81
4303 Historical Studies, 78
```

Figura 2 - Estratto dell'output della Query 1su MongoDB

Query 2

La seconda query effettuata consiste nel contare per ogni anno quante volte si ripresentava un determinato argomento. Per fare ciò abbiamo definito una pipeline che facesse il $match su Reaserch Organization, per poi proseguire facendo un'operazione di $unwind sul campo Fields of Reaserch. Dopo aver raggruppato per start year e Fields of Reaserch, viene eseguito un conteggio degli argomenti di ricerca presenti per ogni anno.

```
1. def countByYear():
2.     db = client.unina
3.     docu = db.docu
4.     pipeline = [
5.         {"$match": {"Research Organization":
6.                     {"$in": ["University of Naples Federico II"]}}},
7.         {'$unwind': '$Fields of Research'},
8.         {'$group':
9.                     {'_id': {'Start Year': '$Start Year', 'Topic': '$Fields of
   Research'}
10.                         , 'count': {'$sum': 1}}},
11.        {'$sort': {'_id': 1}}
12.    ]
13.    result = docu.aggregate(pipeline)
14.    for doc in result:
15.        print(f"Start Year: {doc['_id']}, count {doc['count']}")
```

```
Start Year: {'Start Year': '1980', 'Topic': '40 Engineering'}, count 1
Start Year: {'Start Year': '1980', 'Topic': '49 Mathematical Sciences'}, count 1
Start Year: {'Start Year': '1980', 'Topic': '4902 Mathematical Physics'}, count 1
Start Year: {'Start Year': '1980', 'Topic': '51 Physical Sciences'}, count 1
Start Year: {'Start Year': '1980', 'Topic': '5106 Nuclear and Plasma Physics'}, count 1
Start Year: {'Start Year': '1985', 'Topic': '30 Agricultural, Veterinary and Food Sciences'}, count 1
Start Year: {'Start Year': '1985', 'Topic': '3004 Crop and Pasture Production'}, count 1
Start Year: {'Start Year': '1985', 'Topic': '3008 Horticultural Production'}, count 1
Start Year: {'Start Year': '1986', 'Topic': '30 Agricultural, Veterinary and Food Sciences'}, count 1
Start Year: {'Start Year': '1986', 'Topic': '31 Biological Sciences'}, count 8
Start Year: {'Start Year': '1986', 'Topic': '3101 Biochemistry and Cell Biology'}, count 3
Start Year: {'Start Year': '1986', 'Topic': '3106 Industrial Biotechnology'}, count 4
Start Year: {'Start Year': '1986', 'Topic': '3108 Plant Biology'}, count 1
Start Year: {'Start Year': '1986', 'Topic': '40 Engineering'}, count 1
Start Year: {'Start Year': '1986', 'Topic': '4002 Automotive Engineering'}, count 1
```

Figure 3 - Estratto dell'output della Query 2 su MongoDB

Neo4J

Setup

Per Neo4J abbiamo scelto di installare l'applicativo localmente alle nostre macchine. Una volta fatto ciò, si è effettuata l'operazione di import a partire dal dataset preprocessato come precedentemente specificato, che è stato spostato nella cartella /import del nostro Progetto di Neo4J.

```
LOAD CSV WITH HEADERS FROM 'file:///data_mongo.csv' AS ROW
FIELDTERMINATOR ';'

MERGE (paper:Paper
            {id:ROW.GrantID,
            name:ROW.GrantID,
            startYear:COALESCE(ROW.StartYear, 0),
            endYear:COALESCE(ROW.EndYear, 0),
            funding:COALESCE(ROW.FundingAmountinEUR, 0)
            })
WITH paper,
     SPLIT(ROW.ResearchOrganization, ';') AS orgs,
     SPLIT(ROW.FieldsofResearch, ';') AS topics

FOREACH (u IN orgs |
     MERGE (uni:Universita {name: TRIM(u)})
     MERGE (paper)-[:WRITTENBY]->(uni)
)

FOREACH (topic IN topics |
     MERGE (t:Topic {name: TRIM(topic)})
     MERGE (paper)-[:ABOUT]->(t)
)
```

Questo codice esegue una serie di operazioni per creare nodi e relazioni nel grafo Neo4J. Il comando MERGE viene utilizzato per creare il nodo paper (se questo non esiste già), utilizzando come attributi le colonne del file CSV di input, facendo attenzione a mettere 0 nelle righe che presentavano un valore NULL grazie al comando COALESCE.

Per ogni riga del CSV viene generato un nodo Paper, se non esistono già vengono creati i nodi per tutte le Universita e tutti i Topic associati al Paper appena creato. Si esegue uno split sui campi Research organization e Fields of Research, andando poi ad iterare. Infine, si creano le relazioni tra il nodo Paper ed i nodi di tipo Universita e Topic, rispettivamente WRITTENBY e ABOUT.

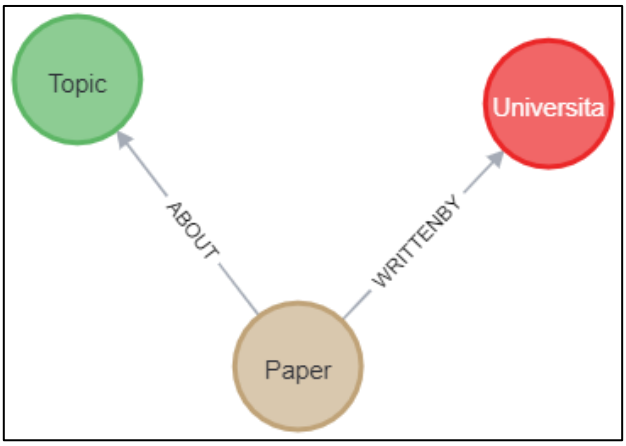

Figura 4 – Grafo rappresentante lo schema del Database di Neo4J

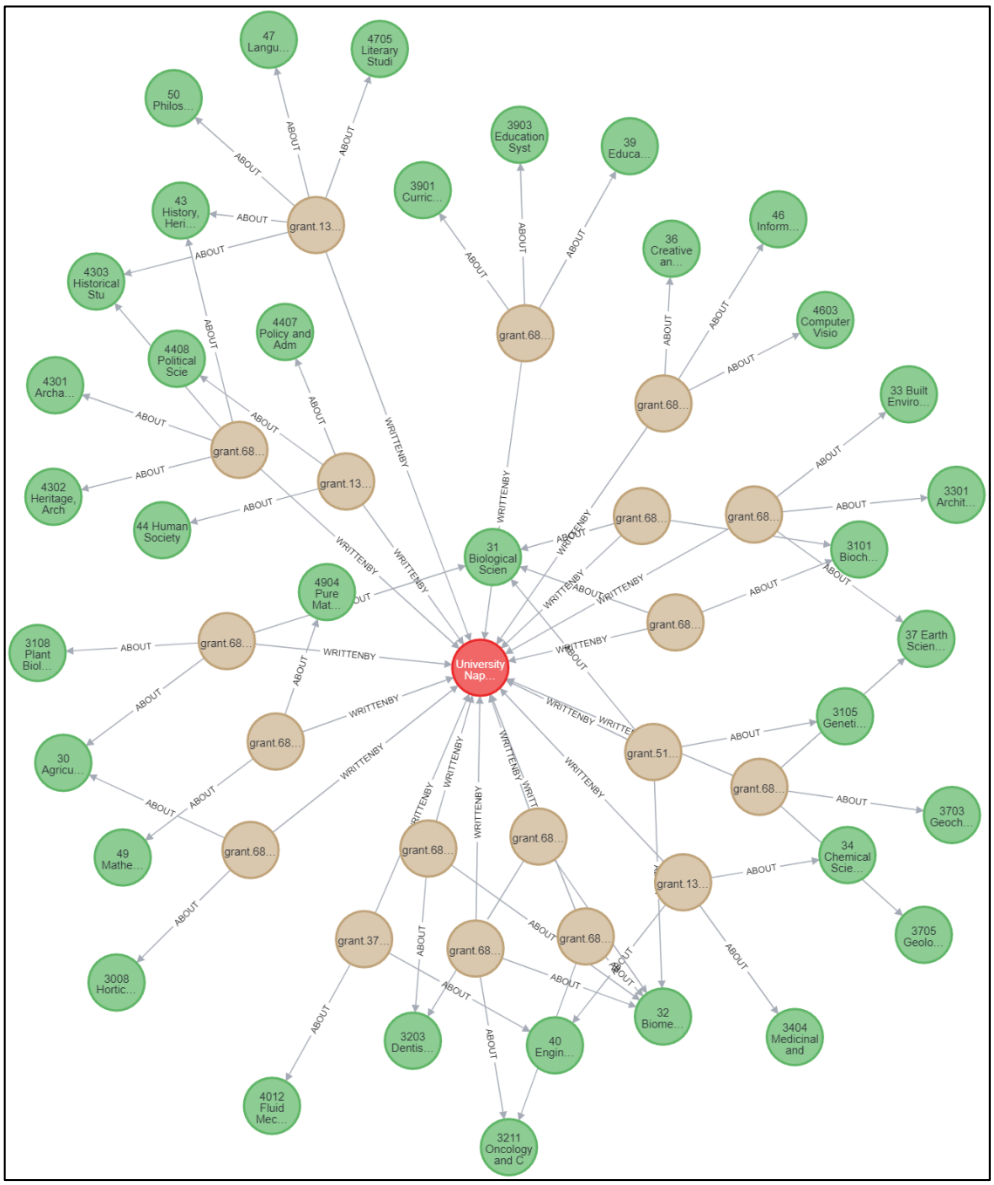

Figura 5 - Visualizzazione del grafo dei progetti in cui è coinvolta la Federico II e i topic connessi

u	w	p	ab	t
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6855738",startYear: "2001",funding: "42349",id: "grant.6855738",endYear: "2003"})	[:ABOUT]	(:Topic {name: "3203 Dentistry"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6855738",startYear: "2001",funding: "42349",id: "grant.6855738",endYear: "2003"})	[:ABOUT]	(:Topic {name: "32 Biomedical and Clinical Sciences"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6858700",startYear: "2010",funding: "703829",id: "grant.6858700",endYear: "2013"})	[:ABOUT]	(:Topic {name: "32 Biomedical and Clinical Sciences"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6858700",startYear: "2010",funding: "703829",id: "grant.6858700",endYear: "2013"})	[:ABOUT]	(:Topic {name: "3211 Oncology and Carcinogenesis"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6852425",startYear: "2010",funding: "450980",id: "grant.6852425",endYear: "2013"})	[:ABOUT]	(:Topic {name: "3703 Geochemistry"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6852425",startYear: "2010",funding: "450980",id: "grant.6852425",endYear: "2013"})	[:ABOUT]	(:Topic {name: "3705 Geology"})
(:Universita {name: "University of Naples Federico II"})	[:WRITTENBY]	(:Paper {name: "grant.6852425",startYear: "2010",funding: "450980",id: "grant.6852425",endYear: "2013"})	[:ABOUT]	(:Topic {name: "37 Earth Sciences"})

Figura 6 - Risultato tabellare del grafo in figura 5

```
1. MATCH(u:Universita)<-[w:WRITTENBY]-(p:Paper)-[ab:ABOUT]->(t:Topic)
2. WHERE u.name="University of Naples Federico II"
3. RETURN u,w,p,ab,t
4. LIMIT 50
```

Con questo codice possiamo visualizzare il grafo di tutti i progetti nei quali è coinvolta la Federico II e tutti i topic ad essi connessi, come visibile in figura 5. In questo caso, data la grossa mole di dati, ci si è limitati a mostrare solo i primi 50 elementi. Nella tabella in figura 6 è mostrato l'output sotto forma tabellare, possiamo notare ad esempio che le prime due righe riguardano lo stesso Paper, ma sono replicate poiché il Paper è collegato a due Topic diversi.

Query 1

Come per quanto già fatto in MongoDB, la query effettuata consiste nel contare le occorrenze di ogni topic nei progetti della Federico II. Il codice scritto è il seguente:

```
1. MATCH(u:Universita)<-[w:WRITTENBY]-(p:Paper)-[ab:ABOUT]->(t:Topic)
2. WHERE u.name="University of Naples Federico II"
3. RETURN COUNT(t) AS COUNT, t.name AS topic
4. ORDER BY COUNT DESC
```

Con questo codice, viene inizialmente effettuato un MATCH andando a cercare nel grafo un nodo università (u) collegato tramite la relazione WRITTENBY ad un nodo Paper (p); tramite questo nodo p saranno poi trovati i nodi di tipo Topic ad esso collegati tramite la relazione ABOUT.

La clausola WHERE specifica la condizione di filtro, cioè che il nodo u sia University of Naples Federico II.

Con il comando RETURN andiamo poi a specificare gli attributi che desideriamo visualizzare nei risultati della query. Ora ci interessa contare le occorrenze di ogni *Field of Reaserch(Topic)* tra tutti i Paper, per farlo utilizziamo il comando COUNT(t) seguito da t.name, l'operazione di raggruppamento è svolta automaticamente dalla funzione RETURN su tutto quello che non è aggregazione, quindi sul campo t.name.

L'output è visibile in figura 7.

count	topic
581	"31 Biological Sciences"
540	"32 Biomedical and Clinical Sciences"
489	"40 Engineering"
248	"34 Chemical Sciences"
211	"3101 Biochemistry and Cell Biology"
164	"46 Information and Computing Sciences"
157	"30 Agricultural, Veterinary and Food Sciences"
153	"44 Human Society"
135	"37 Earth Sciences"

ming 189 records in less than 1 ms and completed after 16 ms.

Figura 7 - Estratto dell'output della Query 1 in Neo4J

Query 2

Anche in Neo4J la seconda query consiste nel contare per ogni anno quante volte si ripresentava un determinato argomento.

È bastato semplicemente fare un match tra i nodi Paper scritti dall'università "University of Naples Federico II" e i topic collegati, per poi effettuare un'operazione di return degli start year dei paper e dei topic collegati; una volta fatto ciò si è eseguito un conteggio.

```
1. MATCH (u:Universita)<-[w:WRITTENBY]-(p:Paper)-[:ABOUT]->(t:Topic)
2. WHERE u.name="University of Naples Federico II"
3. RETURN p.startYear AS startYear, t.name AS topic, COUNT(*) AS COUNT
4. ORDER BY startYear, COUNT DESC
```

L'output è visibile in figura 8.

startYear	topic	count
"1980"	"5106 Nuclear and Plasma Physics"	1
"1980"	"49 Mathematical Sciences"	1
"1980"	"51 Physical Sciences"	1
"1980"	"40 Engineering"	1
"1980"	"4902 Mathematical Physics"	1
"1985"	"3008 Horticultural Production"	1
"1985"	"3004 Crop and Pasture Production"	1
"1985"	"30 Agricultural, Veterinary and Food Sciences"	1
"1986"	"31 Biological Sciences"	8
"1986"	"3106 Industrial Biotechnology"	4

Figure 8 - Estratto dell'output della Query 2 in Neo4J

Grafici

Per meglio visualizzare l'elaborazione effettuata nella query 1, sono stati realizzati due grafici: il primo che rappresenta un istogramma dei principali argomenti dei progetti della Federico II e il secondo che rappresenta un diagramma a torta sulla frequenza dei vari argomenti