Big Data Analytics

Attività n. 3

Marta Caffagnini
Matricola: 168007
307930@studenti.unimore.it

01/12/2021

Neo4j

Indice

1 Esempio di applicazione reale di Neo4j **2**

2 Presentazione dei dati **2**

3 Interrogazioni dello use case **7**
3.1 Query 1 7
3.2 Query 2 7
3.3 Query 3 7

4 Modifica del grafo a supporto di interrogazioni **8**
4.1 Query 4 8
4.2 Query 5 9

5 Altre interrogazioni **11**
5.1 Query 6 11
5.2 Query 7 12
5.3 Query 8 12
5.4 Query 9 12

1. Esempio di applicazione reale di Neo4j

Lo use case scelto è il database “Recipes” trovato al link:
https://neo4j.com/developer/exampledata/#guide-examples

Le ricette contenute nel grafo fanno parte della raccolta BBC GoodFood.

BBC GoodFood è un food magazine e sito web sul cibo che riporta varie ricette della tradizione britannica e non.

2. Presentazione dei dati

Una volta inseriti tutti i dati, il seguente comando permette di visualizzare il metagrafo e vedere i tipi di nodi e relazioni del grafo.

```
CALL db.schema.visualization()
```

Il grafo è costruito intorno a Recipe.

Le Recipes sono in relazione con altri oggetti. Una ricetta contiene ingredienti (Ingredients), può far parte di una Collezione, è scritta da un autore (Author), può far parte di una certa dieta (DietType), e ha determinate parole chiave (Keywords).

Le lable, cioè i tipi di nodo del grafo vengono ritornate dal comando:

```
CALL db.labels()
```

"label"
"Recipe"
"Author"
"Ingredient"
"Keyword"
"DietType"
"Collection"

Il numero di nodi totale è:

```
MATCH (n)
RETURN count(n)
```

"count(n)"
33411

Le relazioni totali sono:

```
MATCH ()-[r]->()
RETURN count(*)
```

"count(*)"
287056

Numero di nodi per ogni lable:

```
MATCH (n)
RETURN labels(n) AS NodeType, count(n) AS NumberOfNodes;
```

"NodeType"	"NumberOfNodes"
["Recipe"]	11634
["Author"]	303
["Ingredient"]	3077
["DietType"]	12
["Collection"]	1049
["Keyword"]	17336

Il nodo di tipo Recipe ha le seguenti proprietà, restituite dal comando:
`MATCH (recipe:Recipe) RETURN keys(recipe) LIMIT 1;`

keys(recipe)
["description", "skillLevel", "id", "cookingTime", "preparationTime", "name"]

- "preparationTime": il tempo di preparazione della ricetta che come valore ha un intero che indica il tempo in secondi.
- "name": nome della ricetta.
- "description": la descrizione della ricetta è una stringa di testo.
- "id": è il codice univoco che identifica la ricetta ed è una stringa.
- "skillLevel": il livello di difficoltà della ricetta. Il suo valore è una stringa e può essere "Easy", "A challenge" "more effort"
- "cookingTime": il tempo di cottura e il suo valore è un intero.

Esempio di un nodo Recipe: Gnocchi bolognese with spinach


```
{
  "identity": 5,
  "labels": [
    "Recipe"
  ],
  "properties": {
"name": "Gnocchi bolognese with spinach",
"preparationTime": 300,
"description": "A filling, cheesy bake that is great for lunch the
                next - if there's any left!",
"id": "102870",
"skillLevel": "Easy",
```

```
  "cookingTime": 1800
    }
  }
```

Il nodo di tipo Ingredient ha solo la proprietà "name" il cui valore è una stringa che riporta il nome dell'ingrediente.

Esempio di un nodo Ingredient: couscous

```
{
  "identity": 11937,
  "labels": [
    "Ingredient"
  ],
  "properties": {
"name": "couscous"
  }
}
```

Il nodo di tipo Author ha solo la proprietà "name" il cui valore è una stringa che riporta il nome dell'autore.

Esempio di un nodo Author: Tony Tobin

```
{
  "identity": 11634,
  "labels": [
    "Author"
  ],
  "properties": {
"name": "Tony Tobin"
  }
}
```

Il nodo di tipo Collection ha la proprietà "name" il cui valore è una stringa che riporta il nome della collezione.

Esempio di un nodo Collection: Chocolate cake

```
{
  "identity": 38734,
  "labels": [
    "Collection"
  ],
  "properties": {
"name": "Chocolate cake"
```

```
  }
}
```

Il nodo di tipo DietType ha la proprietà "name" il cui valore è una stringa che riporta il nome della dieta.

Esempio di un nodo DietType: Low-calorie

```
{
  "identity": 38723,
  "labels": [
    "DietType"
  ],
  "properties": {
"name": "Low-calorie"
  }
}
```

Il nodo di tipo Keyword ha la proprietà "name" il cui valore è una stringa che riporta il nome della parola chiave.

Esempio di un nodo Keyword: Mozzarella

```
{
  "identity": 39784,
  "labels": [
    "Keyword"
  ],
  "properties": {
"name": "Mozzarella"
  }
}
```

Il nodo Recipe è in relazione con gli altri tipi di nodo.

Il nodo di tipo Ingredient è in relazione con Recipe attraverso la relationship di tipo "CONTAINS_INGREDIENT".

Il nodo di tipo Author è in relazione con Recipe attraverso la relazione di tipo "WROTE". Author è il nodo di partenza e Recipe il nodo di arrivo.

Il nodo di tipo Collection è in relazione con Recipe attraverso la relazione di tipo "COLLECTION".

Il nodo di tipo DietType è in relazione con Recipe attraverso la relazione di tipo "DIET_TYPE".

Il nodo di tipo Keyword è in relazione con Recipe attraverso la relazione di tipo "KEYWORD".

3. Interrogazioni dello use case

3.1 Query 1

- Elencare le ricette che contengono l'ingrediente 'chilli'.

```
MATCH (r:Recipe)
WHERE (r)-[:CONTAINS_INGREDIENT]->(:Ingredient {name: 'chilli'})
RETURN r.name AS recipe,
       [(r)-[:CONTAINS_INGREDIENT]->(i) | i.name]  AS ingredients
```

L'ultima riga della query permette di indicare un altro pattern per poi selezionare il nome degli ingredienti della ricetta e inserirli in un array.

3.2 Query 2

- Elencare le ricette che hanno come ingredienti chilli e prawn.

```
:param ingredients => ['chilli', 'prawn']
```

```
MATCH (r:Recipe)
WHERE all(i in $ingredients WHERE exists(
  (r)-[:CONTAINS_INGREDIENT]->(:Ingredient {name: i})))
RETURN r.name AS recipe,
```

Il comando :param ingredients=> ['chilli', 'prawn'] definisce un parametro chiamato ingredients che verrà inviato alla query.

La parte a destra della freccia viene valutata come codice in linguaggio Cypher con una RETURN implicita in testa.

Il comando WHERE prende tutte le Recipe che tra gli ingredienti hanno sia chilli che prawn.

3.3 Query 3

- Elencare le ricette che hanno come ingredienti coconut milk e rice, ma non hanno gli allergeni egg e milk.

```
:param allergens =>   ['egg', 'milk'];
:param ingredients => ['coconut milk', 'rice'];

MATCH (r:Recipe)
WHERE all(i in $ingredients WHERE exists(
  (r)-[:CONTAINS_INGREDIENT]->(:Ingredient {name: i})))
AND none(i in $allergens WHERE exists(
  (r)-[:CONTAINS_INGREDIENT]->(:Ingredient {name: i})))
RETURN r.name AS recipe,
       [(r)-[:CONTAINS_INGREDIENT]->(i) | i.name] AS ingredients
```

Nelle prime due righe vengono creati i parametri allergens e ingredients con gli ingredienti corrispondenti.

Nel comando WHERE si impone che le Recipe abbiano entrambi gli ingredienti presenti nella variabile ingredients e contemporaneamente non abbia nessuno degli ingredienti presenti in allergens.

4. Modifica del grafo a supporto di interrogazioni

4.1 Query 4

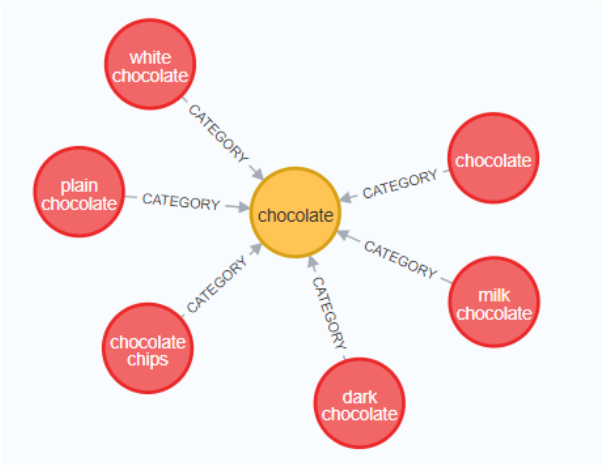
- Elencare ricette che contengono del cioccolato.

Creo un nuovo tipo di nodo. IngredientCategory è il lable del nuovo nodo che si mette in relazione con altri nodi di tipo Ingredient con la relationship :CATEGORY. Creo un nodo con attributo name:'chocolate' e lo metto in relazione con ingredienti che sono vari tipi di cioccolato. I nomi di questi ingredienti sono elencati nel parametro chocolate.
FOREACH viene utilizzato per modificare il grafo. Per ogni nome contenuto nel parametro chocolate viene creato un ingrediente, poi con MERGE viene verificata la presenza dell'IngredientCategory chocolate che viene messa in relazione con l'ingrediente.

```
:param chocolate=> ['dark chocolate', 'milk
chocolate','chocolate','white chocolate','plain chocolate', 'chocolate
chips','chocolate sauce', 'hot chocolate powder'];

FOREACH (ingredient IN $chocolate |
  MERGE(i:Ingredient {name: ingredient})
  MERGE (chocolate:IngredientCategory {name: 'chocolate'})
  MERGE (i)-[:CATEGORY]->(chocolate)

)
```



Interrogazione:

```
MATCH
(r:Recipe)-[:CONTAINS_INGREDIENT]->(i)-[:CATEGORY]->(cat:IngredientCate
gory{name:'chocolate'})

RETURN r.name
```

4.2 Query 5

- Elencare 50 ricette senza spezie.

Per rispondere a questa interrogazione inizialmente viene creato un parametro il cui valore è una lista di nomi di ingredienti che sono spezie.

```
:param spices => ['ground cinnamon', 'cinnamon','star
anise','cumin','chilli','red chilli', 'chilli flakes','chilli
powder','chilli sauce','green chilli','jalapeno chilli','chillies',
'mixed spice','coriander','green pepper','cayenne pepper','curry
powder','paprika','spice','Romano pepper'];
```

In seguito viene creato un nuovo nodo con attributo name:'spice' e come lable un nuovo tipo: IngredientCategory. Questo nuovo tipo di nodo si mette in relazione con gli ingredienti che fanno parte della categoria indicata nell'attributo name. Questa relazione si chiama :CATEGORY.

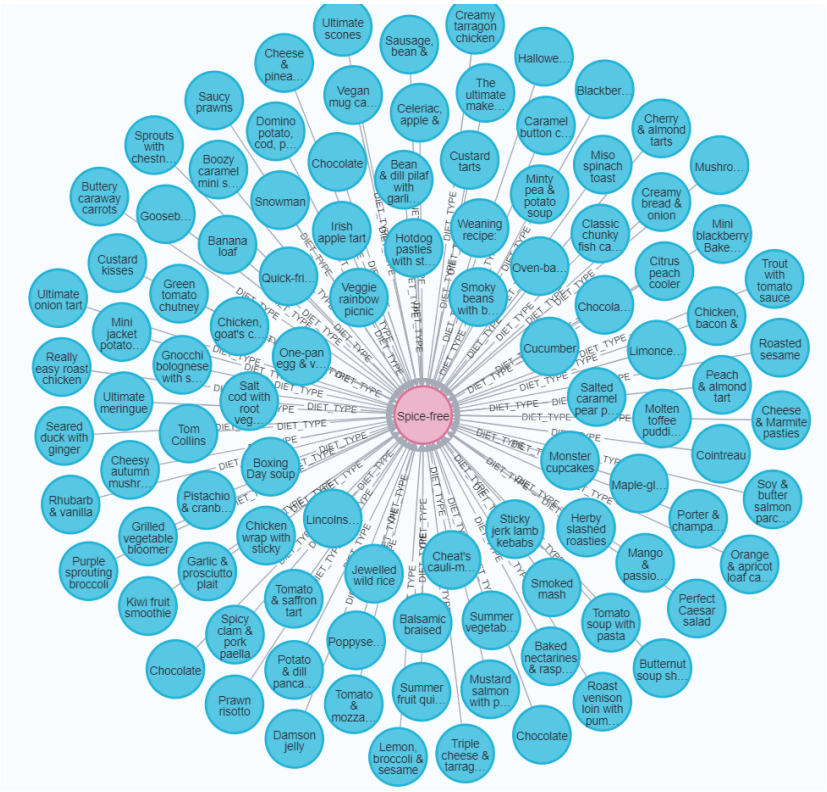
```
FOREACH (ingredient IN $spices |
  MERGE (i:Ingredient {name: ingredient})
  MERGE (spice:IngredientCategory {name:'spice'})
  MERGE (i)-[:CATEGORY]->(spice)
```

)

Per ogni ingrediente indicato nel parametro spices viene creata una relazione con il nodo spice di tipo :IngredientCategory.
Di seguito il grafo risultante.

Come ultima modifica viene creato un nuovo nodo del tipo DietType con attributo name: "Spice-free". Questo nodo viene poi messo in relazione di tipo :DIET_TYPE con tutte le ricette che non contengono neanche un ingrediente della categoria ‘spice’.

```
CREATE (Spice_free:DietType {name:"Spice-free"});
MATCH (r:Recipe)
MATCH (d:DietType{name:'Spice-free'})
WHERE not exists(
(r)-[:CONTAINS_INGREDIENT]->(:Ingredient)-[:CATEGORY]->(:IngredientCat
egory{name:'spice'})
)
MERGE (r)-[:DIET_TYPE]->(d);
```



Interrogazione:

```
MATCH (r:Recipe)-[:DIET_TYPE]->(d:DietType{name:'Spice-free'})
RETURN r.name
LIMIT 50;
```

5. Altre interrogazioni

5.1 Query 6

- Elencare i 5 autori che hanno scritto più ricette con l'ingrediente butter

```
MATCH
(a:Author)-[:WROTE]->(r:Recipe)-[:CONTAINS_INGREDIENT]->(i:Ingredient{n
ame:"butter"})
RETURN a.name, count(*) AS number
ORDER BY number DESC
LIMIT 5
```

5.2 Query 7

- Elenca 50 ricette che si possono fare con tempo di preparazione minore di un'ora e tempo di cottura minore di 10 minuti. Elencare quali ingredienti occorrono per ogni ricetta.

```
MATCH (r:Recipe)-[:CONTAINS_INGREDIENT]->(i:Ingredient)
WHERE r.preparationTime<3600
RETURN r.name AS Ricetta, collect(i.name) AS Ingredienti
LIMIT 50
```

5.3 Query 8

- Elencare ricette di natale per intolleranti al lattosio e allergici alle castagne

```
MATCH (r:Recipe)-[:COLLECTION]->(c:Collection{name:"Christmas
sprouts"})
WHERE none(i in ["milk","chestnut"] WHERE exists(
  (r)-[:CONTAINS_INGREDIENT]->(:Ingredient {name: i})))
RETURN r.name AS Ricetta, [(r)-[:CONTAINS_INGREDIENT]->(i) | i.name] AS
Ingredienti
```

5.4 Query 9

- Elencare le 50 ricette semplici e più veloci (cookingTime+ preparationTime)

```
MATCH (r:Recipe{skillLevel:"Easy"})
RETURN r.name, r.cookingTime+r.preparationTime AS Time
ORDER BY Time
LIMIT 50
```