Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

PROJECTE NEO4J

Bases de Dades No Relacionals

Introducció

Aquest informe documenta la resolució d'un projecte utilitzant Neo4j per gestionar dades de padrons municipals. Es descriuen les tasques realitzades, la distribució de tasques entre els membres de l'equip, així com els scripts i consultes en Cypher per importació de dades, resolució d'exercicis i anàlisi de grafs.

Adreça del Repositori

L'adreça del repositori que conté tot el material generat per aquest informe és:

https://github.com/Dakuur/Neo4j_1666540

Treball en Equip

David Morillo	- **Importació de les dades**
Albert Guillaumet	- **Consultes de Cypher**
Adrià Muro	- **Redacció de l'informe**
Lucía Garrido	- **Analítica de Grafs**

Cada membre ha treballat principalment en la seva tasca assignada, reflectint-se en els commits del repositori del projecte, sense deixar de revisar i ajudar amb les tasques d'altres companys.

***Important**: Les comandes Cypher fetes servir es troben també en format .txt en el repositori de GitHub, separades per exercicis.

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

Exercici 1: Importació de Dades en la BD de Neo4j

Hem importat dades en la base de dades de Neo4j mitjançant un script en Cypher que genera nodes, relacions i afegeix les característiques corresponents. Ens hem assegurat que l'execució de l'script dues vegades no dupliqui les dades, mitjançant l'ús de la clàusula “MERGE” i indexos. A continuació s’explica cada fase de la creació de la db.

CREATE INDEX house_id_index IF NOT EXISTS FOR (n:House) ON (n.id);
CREATE INDEX individual_id_index IF NOT EXISTS FOR (n:Individual) ON (n.id);

Es creen indexos per als nodes House i Individual basats en el camp id per millorar el rendiment de les consultes. Es fa ús de IF NOT EXISTS per evitar errors si els indexos ja existeixen.

```
LOAD CSV WITH HEADERS FROM
'https://docs.google.com/spreadsheets/d/e/2PACX-1vT0ZhR6BSO_M72JEmxXKs6GLuOwxm_Oy-0UruLJeX8_R04KAclCuvrwn2OENQhtuvddU5RSJSclHRJf/pub?output=csv'
AS row
WITH row, row.Municipi + "_" + row.Any_Padro + "_" + row.Id_Llar AS house_id
MERGE (n:House {id: house_id})
SET n.year_padron = row.Any_Padro,
    n.number = row.Numero,
    n.municipality = row.Municipi,
    n.house_id = row.Id_Llar,
    n.street = row.Carrer;
```

Aquesta fase carrega les dades de les cases des d'un fitxer CSV. Es genera un identificador únic per a cada casa combinant el municipi, l'any del padró i l'ID de la llar (house_id). Es fa servir MERGE per evitar duplicats i SET per establir les propietats dels nodes House.

```
LOAD CSV WITH HEADERS FROM
'https://docs.google.com/spreadsheets/d/e/2PACX-1vTfU6oJBZhmhzzkV_0-avABPzHTdXy8851ySDbn2gq32WwaNmYxfiBtCGJGOZsMgCWjzlEGX4Zh1wqe/pub?output=csv'
AS row
MERGE (n:Individual {id: row.Id})
SET n.year = row.Year,
    n.name = row.name,
    n.surname = row.surname,
    n.second_surname = row.second_surname;
```

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

Aquesta fase carrega les dades dels individus des d'un fitxer CSV. Utilitza MERGE per crear nodes Individual evitant duplicats, i SET per establir les seves propietats.

```
LOAD CSV WITH HEADERS FROM
'https://docs.google.com/spreadsheets/d/e/2PACX-1vRVOoMAMoxHiGboTjClHo2yT30CCWgVHgocGnVJxiCTgyurtmqCfAFahHajobVzwXFLwhqajz1fqA8d/pub?output=csv'
AS row
MATCH (n1:Individual {id: row.ID_1}), (n2:Individual {id: row.ID_2})
MERGE (n1)-[f:IS_FAMILY]->(n2)
SET f.relationship = row.Relacio,
    f.harmonized_relationship = row.Relacio_Harmonitzada;
```

Aquesta fase carrega les relacions familiars entre individus des d'un fitxer CSV. Els nodes d'individus es troben amb MATCH i es crea una relació IS_FAMILY utilitzant MERGE per evitar duplicats. Les propietats de la relació es configuren amb SET.

```
LOAD CSV WITH HEADERS FROM
'https://docs.google.com/spreadsheets/d/e/2PACX-1vRM4DPeqFmv7w6kLH5msNk6_Hdh1wuExRirgysZKO_Q70L21MKBkDISIyjvdm8shVixI5Tcw_5zCfdg/pub?output=csv'
AS row
WITH row, row.Location + "_" + row.Year + "_" + row.HOUSE_ID AS house_id
MATCH (n:Individual {id: row.IND}), (h:House {id: house_id})
MERGE (n)-[l:LIVES_IN]->(h)
SET l.lives = row.VIU,
    l.location = row.Location,
    l.year = row.Year;
```

Aquesta fase carrega les relacions de residència entre individus i cases des d'un fitxer CSV. Es genera un house_id únic, es troben els nodes Individual i House amb MATCH, i es crea la relació LIVES_IN utilitzant MERGE. Les propietats de la relació es configuren amb SET.

```
LOAD CSV WITH HEADERS FROM
'https://docs.google.com/spreadsheets/d/e/2PACX-1vTgC8TBmdXhjUOPKJxyiZSpetPYjaRC34gmxHj6H2AWvXTGbg7MLKVdJnwuh5bleer7WLUi0Oigl6wc/pub?output=csv'
AS row
MATCH (n1:Individual {id: row.Id_A}), (n2:Individual {id: row.Id_B})
MERGE (n1)-[:SAME_AS]->(n2);
```

Aquesta fase carrega les relacions SAME_AS entre individus des d'un fitxer CSV. Els nodes d'individus es troben amb MATCH i es crea la relació SAME_AS utilitzant MERGE per evitar duplicats.

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

Exercici 2: Consultes Cypher

Hem resolt diverses consultes Cypher per obtenir informació específica dels padrons municipals. A continuació es motraran les diferents consultes juntament amb una captura de pantalla del resultat al executar-les.

1- Per a cada padró (any) de Sant Feliu de Llobregat (SFLL), retorna l'any de padró, el número d'habitants, i la llista de cognoms. Elimina duplicats i "nan".

```
MATCH (h:House {municipality: 'SFLL'})<-[:LIVES_IN]-(i:Individual)
WITH h.year_padron as padro, i
WHERE i.surname IS NOT NULL AND i.surname <> "nan"
RETURN padro as padro, COUNT(DISTINCT i) AS num_habitants, COLLECT(DISTINCT
i.surname) as cognoms
ORDER BY padro
```

	padro	num_habitants	cognoms
1	"1833"	345	["roig", "aloma", "guiu", "casas", "coca", "majo", "parellada°", "canalias", "camprecios", "ribas", "va
2	"1838"	71	["canameras", "gullart", "bonsoms", "sola", "pares", "ramos", "comellas", "molins", "marles", "fabre
3	"1839"	490	["ribas", "illegible", "roig", "parera", "camprubi", "aloma", "majo", "pañella", "caralto", "camprecios",
4	"1878"	2739	["vidal", "sadurni", "pahisa", "anavall", "sarrio", "corrons", "galtes", "amigo", "marti", "castells", "plai
5	"1881"	2999	["esteba", "alsina", "marti", "castells", "casas", "olle", "tort", "sanllehi", "mas", "sala", "nin", "ferran",
6	"1889"	3109	["mas", "sala", "deu", "marles", "ferres", "arbulos", "serralabos", "codinachs", "tutusaus", "pelegri",

2- Retorna totes les aparicions de "miguel estape bofill". Fes servir la relació SAME_AS per poder retornar totes les instancies, independentment de si hi ha variacions lèxiques (ex. diferents formes d'escriure el seu nom/cognoms). Mostra la informació en forma de taula: el nom, la llista de cognoms i la llista de segon cognom (elimina duplicats).

```
MATCH (i:Individual)-[:SAME_AS]->(in:Individual)
WHERE (in.name = "miguel" AND in.surname = "estape" AND in.second_surname = "bofill")
WITH i
```

```
RETURN i.name as nom,
    COLLECT(DISTINCT i.surname) as cognoms,
    COLLECT(DISTINCT i.second_surname) as segons_cognoms
ORDER BY nom
```

	nom	cognoms	segons_cognoms
1	"miguel"	["estape"]	["bufill", "bofill"]

3- Mostra els fills o filles(només) de "benito julivert". Mostra la informació en forma de taula: el nom, cognom1, cognom2, i tipus de relació. Ordena els resultats alfabèticament per nom.

```
MATCH(pare:Individual{name: "benito", surname:"julivert"})-[r:IS_FAMILY]->(fill:Individual)
WHERE r.harmonized_relationship IN ["fill","filla"]
WITH fill,r

RETURN fill.name as nom_fill,
    fill.surname as cognom_fill,
    fill.second_surname as segon_cognom_fill,
    r.harmonized_relationship as tipus_de_relacio

ORDER BY nom_fill
```

	nom_fill	cognom_fill	segon_cognom_fill	tipus_de_relacio
1	"dolores"	"julibert"	"julia"	"filla"
2	"joaquina"	"julibert"	"julia"	"filla"
3	"jose"	"julibert"	"julia"	"fill"
4	"juan"	"julibert"	"julia"	"fill"
5	"magdalena"	"julibert"	"julia"	"filla"
6	"martin"	"julibert"	"julia"	"fill"

4- Mostreu les famílies de Castellví de Rosanes amb més de 3 fills. Mostreu el nom i cognoms del cap de família i el nombre de fills. Ordeneu-les pel nombre de fills fins a un límit de 20, de més a menys.

```
MATCH (h:House{municipality:"CR"})<-[:LIVES_IN]-(p:Individual)
WITH p, h
MATCH (p)-[r:IS_FAMILY]->(fill:Individual)
WHERE r.harmonized_relationship IN ["fill","filla"]
WITH p,COUNT(fill) AS numero_fills
WHERE numero_fills > 3

RETURN p.name as nom_cap,
    p.surname as cognom1_cap,
    p.second_surname as cognom2_cap,
    numero_fills as num_fills
ORDER BY num_fills DESC
LIMIT 20
```

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

	nom_cap	cognom1_cap	cognom2_cap	num_fills
1	"pablo"	"astruch"	"julia"	7
2	"jose"	"olle"	"domenech"	6
3	"benito"	"julivert"	"parera"	6
4	"jose"	"canals"	"olle"	6
5	"pedro"	"bargallo"	"ilegible"	6
6	"jose"	"canals"	"mila"	6
7	"jose"	"rafuls"	"mila"	5
8	"jaime"	"jarrey"	"ilegible"	5
9	"pablo"	"bargallo"	"armangol"	5
10	"francisco"	"aregay"	"rigol"	5
11	"pablo"	"canals"	"llimona"	4
12	"ramon"	"canals"	"amat"	4
13				

13	"jaime"	"gallofre"	"bartran"	4
14	"tomas"	"parera"	"roig"	4
15	"juan"	"julibert"	"parera"	4
16	"estevan"	"gallofre"	"bertran"	4
17	"cristobal"	"olle"	"rabantos"	4
18	"pedro"	"farres"	"rigol"	4

5- Per cada padró/any de Sant Feliu de Llobregat, mostra el carrer amb menys habitants i el nombre d'habitants en aquell carrer. Fes servir la funció min() i CALL per obtenir el nombre mínim d'habitants. Ordena els resultats per any de forma ascendent.

```
MATCH (h:House {municipality: 'SFLL'})<-[:LIVES_IN]-(i:Individual)
WITH h.year_padron AS any, h.street AS carrer, COUNT(i) AS num_habitants
ORDER BY any, num_habitants
WITH any, COLLECT({carrer: carrer, num_habitants: num_habitants}) AS carrers_per_any
RETURN any, carrers_per_any[0].carrer AS carrer_amb_menys_habitants,
carrers_per_any[0].num_habitants AS num_habitants
ORDER BY any;
```

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

	any	carrer_amb_menys_habitants	num_habitants
1	"1833"	"carrtera de la part de molins de rey"	5
2	"1838"	"carretera de barna"	30
3	"1839"	"casas del 3onmany"	3
4	"1878"	"carrretera"	2
5	"1881"	"Carretera"	5
6	"1889"	"s n antonio"	1

Lucía Garrido, Albert Guillaumet, David Morillo i Adrià Muro
9.06.2024

Exercici 3: Analítica de Grafs

Hem realitzat una anàlisi de grafs per entendre millor l'estructura de les dades, incloent l'estudi de components connexes i la semblança entre nodes.

ESTUDI DE LES COMPONENTS CONNEXES

1 - Mostra, en forma de taula, les 10 components connexes més grans (ids i mida).

```
CALL gds.wcc.stream({
  nodeProjection: ['Individual', 'House'],
  relationshipProjection: {
    LIVES_IN: {
      type: 'LIVES_IN',
      orientation: 'UNDIRECTED'
    }
  }
})
YIELD nodeId, componentId
WITH componentId, COUNT(nodeId) AS size
RETURN componentId, size
ORDER BY size DESC
LIMIT 10;
```

Aquesta consulta busca les 10 components connexes més grans del graf que representa les connexions entre nodes Individuals i Habitatges. No obstant, els quatre integrants hem tingut problemes amb el mode “stream”, encara així, considerem correcte el codi.

2 - Per cada municipi i any el nombre de parelles del tipus: (Individu)—(Habitatge).

```
MATCH (i:Individual)-[:LIVES_IN]->(h:House) RETURN h.municipality AS municipi, h.year_padron AS any, COUNT(*) AS num_parelles ORDER BY municipi, any;
```

1	"CR"	"1866"	337
2	"SFLL"	"1833"	1433
3	"SFLL"	"1838"	287
4	"SFLL"	"1839"	1946
5	"SFLL"	"1878"	2745
6	"SFLL"	"1881"	3000

Aquesta consulta compta el nombre de parelles (Individu)—(Habitatge) per cada municipi i any. El primer valor correspon al municipi, el segon a l'any i el tercer guarda el nombre de parelles.

SEMBLANÇA ENTRE NODES

1 - Determinar els habitatges que són els mateixos al llarg dels anys i afegir una aresta "MATEIX_HAB"

```
MATCH (h1:House), (h2:House)
WHERE h1.house_id = h2.house_id AND h1.year_padron < h2.year_padron
MERGE (h1)-[:MATEIX_HAB]->(h2);
```

Aquesta consulta crea una relació "MATEIX_HAB" entre habitatges que tenen el mateix identificador al llarg dels anys.

2 - Crear un graf en memòria amb els nodes Individu i Habitatge i les relacions VIU, FAMILIA, MATEIX_HAB

```
CALL gds.graph.project(
  'graf_individu_habitatge',
  ['Individual', 'House'],
  {
    LIVES_IN: {
      type: 'LIVES_IN',
      orientation: 'UNDIRECTED'
    },
    IS_FAMILY: {
      type: 'IS_FAMILY',
      orientation: 'UNDIRECTED'
    },
    MATEIX_HAB: {
      type: 'MATEIX_HAB',
      orientation: 'UNDIRECTED'
    }
  }
);
```

nodeProjection	relationshipProjection	graphName	nodeCount	relationshipCount	projectMillis
1 { "House": { "label": "House", "properties": { } }, "Individual": { "label": "Individual",	{ "MATEIX_HAB": { "aggregation": "DEFAULT", "orientation": "UNDIRECTED", "indexInverse": false, "properties": { },	"graf_individu_habitatge"	21288	71510	151

Aquesta operació crea un graf en memòria amb nodes Individuals i Habitatges, i inclou les relacions VIU, FAMILIA i MATEIX_HAB.

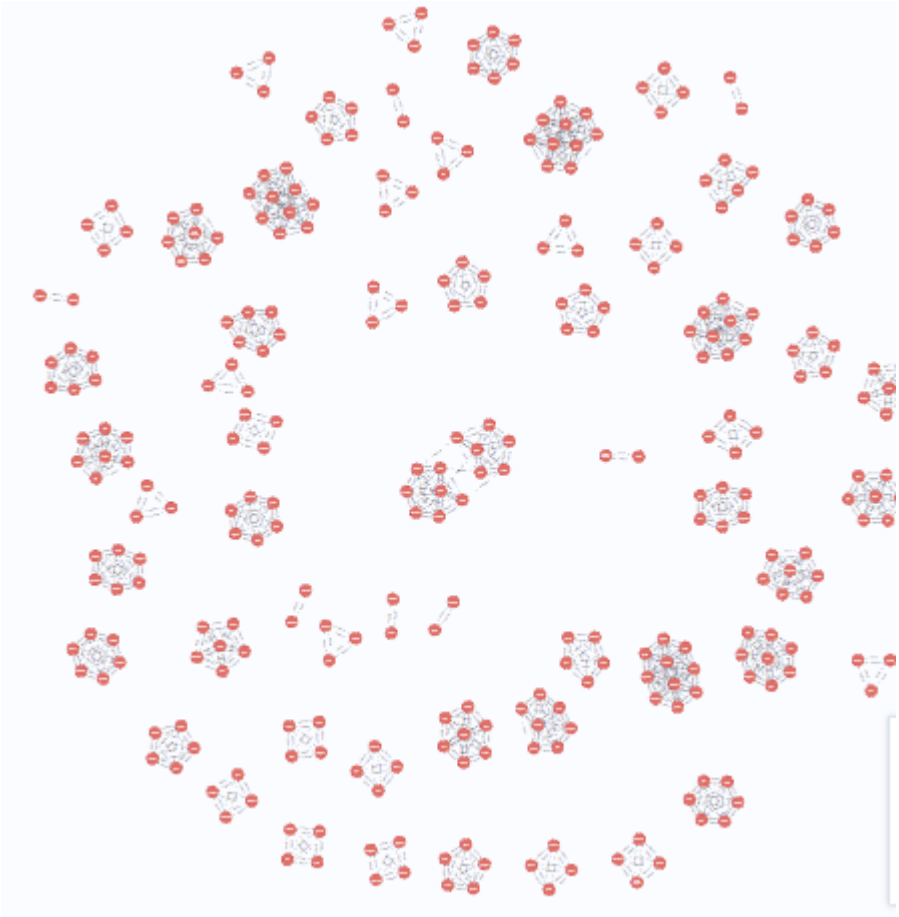
Amb un node count de 21288 i un relationship count de 71510, el graf resultant és notablement gran, suggerint una ampla i rica interconnexió entre els nodes Individuals i Habitatges.

El projectmillis, que indica el temps de projecció del graf en memòria, és de 151 mil·lisegons, la qual cosa indica una operació relativament ràpida malgrat la complexitat del graf. Això suggereix una eficient estructuració de dades i un rendiment robust del sistema per a aquesta tasca.

3 - Calcular la similaritat entre els nodes del graf creat

```
CALL gds.nodeSimilarity.write({
  nodeProjection: ['Individual', 'House'],
  relationshipProjection: {
    LIVES_IN: {
      type: 'LIVES_IN',
      orientation: 'UNDIRECTED'
    },
    IS_FAMILY: {
      type: 'IS_FAMILY',
      orientation: 'UNDIRECTED'
    },
    MATEIX_HAB: {
      type: 'MATEIX_HAB',
      orientation: 'UNDIRECTED'
    }
  },
```

```
  writeRelationshipType: 'SIMILAR_TO',
  writeProperty: 'similarityScore'
});
```



Aquesta consulta calcula la similaritat entre nodes del graf creat i escriu els resultats en una nova relació anomenada 'SIMILAR_TO' amb la propietat 'similarityScore'.