

BASES DE DADES NO RELACIONALS  
MATEMÀTICA COMPUTACIONAL I ANALÍTICA DE DADES  
UNIVERSITAT AUTÒNOMA DE BARCELONA

# PROJECTE NEO4J

PADRONS

Bruno Tejedo 1533327  
Arnau Añols 1603271  
Maria Ganduxé 1606755  
Jan Pérez 1562911

## Índex

<b>1</b>	<b>Introducció</b>	<b>2</b>
<b>2</b>	<b>Treball en equip</b>	<b>2</b>
<b>3</b>	<b>Exercici 1</b>	<b>2</b>
<b>4</b>	<b>Exercici 2</b>	<b>4</b>
<b>5</b>	<b>Exercici 3</b>	<b>11</b>
5.1	Estudi de les components connexes (cc) i de l'estructura de les component en funció de la seva mida. . . . .	11

## 1 Introducció

Un grup d'investigadores en demografia històrica i processament de documents vol estudiar l'evolució socio-econòmica d'una població a partir de la informació dels padrons de poblacions. Els padrons són els llistats d'habitants que elabora un municipi on figura la informació la seva informació com noms, cognoms, edat i altres dades personals. En l'actualitat aquesta informació es manté actualitzada constantment gràcies als sistemes informàtics però antigament es recopilava manualment cada 3-5 anys aproximadament.

La informació d'aquests documents, un cop digitalitzats, es processen amb tècniques de visió per computador i s'organitzen en una base de dades relacional. Per construir la base de dades, s'ha extret, de cada padró, la informació de cada habitatge, les persones que hi viuen i la relació de parentesc que hi ha entre ells. De cada habitatge, guarden la adreça completa (carrer, numero, pis), codi postal, barri i població. De cada persona que viu a l'habitatge, es guarda quina relació de parentesc el vincula amb el/la cap de família. A més, necessiten saber la ocupació (si treballa), una estimació dels ingressos (bruts) anuals de cadascun d'ells i l'estat civil. A més, com que els enregistraments poblacionals es repeteixen cada 3-5 anys la informació que es recopila per cada habitatge es va repetint per aquelles famílies que viuen en els mateixos habitatges.

Interessa identificar la informació dels individus al llarg del temps. Aquesta identificació es fa essencialment a partir de les dades personals (nom i cognoms) però en ocasions no és suficient. En aquests casos s'utilitza altres dades, com la data de naixement (inferida a partir de l'edat i l'any de padró), l'ofici o les relacions familiars.

Per poder completar aquesta tasca primer necessitarem importar les dades dels fitxers csv que se'ns han facilitat. A continuació, generarem un script en *cypher* que carregui totes aquestes dades i generi el graf afegint les característiques necessàries.

## 2 Treball en equip

### 3 Exercici 1

**Importa les dades en la BD de Neo4j del projecte. Genera un script en cypher que carregui totes les dades, generi tots els nodes, relacions i afegeixi les característiques allà on toqui.**

Per poder començar el nostre projecte el primer que hem de fer és importar totes les dades que se'ns proporcionen per tal de crear una base de dades amb la qual puguem treballar des del Neo4j. Per fer-ho, generem el següent script en cypher.

1. Creació de restriccions i índexos: Creem les restriccions úniques per assegurar que els nodes d'Habitatge tenen un Id Llar únic i els nodes d'Individu també. Amb això garantim que no es puguin crear nodes duplicats amb els mateixos identificadors.

```
CREATE CONSTRAINT habitatge_unique_id_constraint
```

```
IF NOT EXISTS FOR (h:Habitatge) REQUIRE h.id_llar IS UNIQUE;
CREATE CONSTRAINT individu_unique_id_constraint
IF NOT EXISTS FOR (i:Individu) REQUIRE i.id IS UNIQUE;
```

2. Càrrega de dades d'Habitatge: Amb la comenada LOAD CSV carreguem les dades del fitxer CSV "HABITATGES.CSV". Verifiquem que les columnes Municipi i Id Llar no siguin nul·les. Amb la comanda MERGE agafem els nodes d'Habitatge amb un Id Llar corresponent al valor de la columna Id Llar del fitxer CSV. Finalment amb la comanda SET assignem les característiques del node Habitatge a partir de les dades del fitxer CSV.

```
LOAD CSV WITH HEADERS FROM 'file:///Dades/HABITATGES.csv' AS row
WITH row WHERE row.Municipi IS NOT NULL AND row.Id_Llar IS NOT NULL
MERGE (h:Habitatge {id_llar: toInteger(row.Id_Llar)})
SET h.any_padro = toInteger(row.Any_Padro),
    h.carrer = row.Carrer,
    h.numero = row.Numero;
```

3. Càrrega de dades d'Individu: Fem el mateix que amb el dataset Individu.

```
LOAD CSV WITH HEADERS FROM 'file:///Dades/INDIVIDUAL.csv' AS row
WITH row WHERE row.Id IS NOT NULL
MERGE (i:Individu {id: toInteger(row.Id)})
SET i.year = toInteger(row.Year),
    i.name = row.name,
    i.surname = row.surname,
    i.second_surname = row.second_surname;
```

4. Creació de relació "VIU": Carreguem el fitxer CSV "VIU.CSV" per crear la relació VIU. Verifiquem que les columnes IND i VIU no siguin nul·les. Busquem el node Habitatge amb l'Id Llar corresponent al valor de la columna HOUSE ID del fitxer CSV i el node individu amb l'id corresponent al valor de la columna IND. Finalment fem el mateix merge que a les anteriors.

```
LOAD CSV WITH HEADERS FROM 'file:///Dades/VIU.csv' AS row
WITH row WHERE row.IND IS NOT NULL AND row.VIU IS NOT NULL
MATCH (h:Habitatge {id_llar: toInteger(row.HOUSE_ID)})
MATCH (i:Individu {id: toInteger(row.IND)})
MERGE (i)-[:VIU {location: row.Location, year: toInteger(row.Year)}]->(h);
```

5. Creació de relació "família": Carreguem el fitxer CSV "FAMILIA.CSV" per establir la relació FAMILIA. Verifiquem les columnes no nul·les i busquem el node Individu amb l'id corresponent al valor de la columna ID 1 i el node Individu amb l'id corresponent al valor de la columna ID 2. Amb la comanda SET s'assignen les característiques de la relació família, "relació" "relació harmonitzada".

```
LOAD CSV WITH HEADERS FROM 'file:///Dades/FAMILIA.csv' AS row
WITH row WHERE row.ID_1 IS NOT NULL AND row.ID_2 IS NOT NULL
MATCH (i1:Individu {id: toInteger(row.ID_1)})
MATCH (i2:Individu {id: toInteger(row.ID_2)})
MERGE (i1)-[:FAMILIA {relacio: row.Relacio, relacio_harmonitzada: row.Relacio_Harmonitzada}]
```

6. Creació de relació "same as": Fem el mateix que hem fet amb la relació família.

```
LOAD CSV WITH HEADERS FROM 'file:///Dades/SAME_AS.csv' AS row
WITH row WHERE row.Id_A IS NOT NULL AND row.Id_B IS NOT NULL
MATCH (i1:Individu {id: toInteger(row.Id_A)})
MATCH (i2:Individu {id: toInteger(row.Id_B)})
MERGE (i1)-[:SAME_AS]->(i2);
```

7. Índexos per a consultes eficients: es creen índexos pels atribus carrer dels nodes Habitatge i "name" dels nodes individu per millorar l'eficiència de les consultes que utilitzen aquests atributs.

```
CREATE INDEX habitatge_carrer_index IF NOT EXISTS FOR (h:Habitatge) ON (h.carrer);
CREATE INDEX individu_name_index IF NOT EXISTS FOR (i:Individu) ON (i.name);
```

## 4 Exercici 2

En aquest exercici se'ns demana resoldre 14 consultes.

**Consulta 1:** Del padró de 1866 de Castellví de Rosanes (CR), retorna el número d'habitants i la llista de cognoms, sense eliminar duplicats.

```
1 //EX 1
2 MATCH (i:Individu)-[:VIU {location: 'CR'}]->(h:Habitatge)
3 WHERE i.year = 1866
4 RETURN COUNT(i) AS num_habitants, COLLECT(DISTINCT i.surname) AS cognoms
```

En aquesta consulta primer fem un match per establir el patró de cerca. Seleccionem els nodes d'Individu

que estiguin connectats a nodes d'Habitatge mitjançant una relació de VIU concretant la població que se'ns demana. Amb això ja hem seleccionat els habitants de Castellví de Rosanes. Per corregir l'any 1866 utilitzem la consulta *WHERE i.year = 1866*. Finalment en el return fem un count per a que ens torni el número d'habitants i amb el collect la llista de cognoms. Aquesta consulta ens retorna el següent:

"num_habitants"	"cognoms"
337	["galceran", "olle", "ros", "julia", "vila", "suñol", "julivert", "julibert", "volta", "canals", "farre", "gallofre", "llopart", "anducas", "capellades", "amat", "ilegible", "mitjans", "rios", "gaset", "valls", "farreny", "bargallo", "rusell", "rigol", "dalmases", "canald", "calaf", "llimona", "masana", "gol", "plana", "cadafaly", "domenech", "aregay", "pujadas", "pascual", "astruch", "rafuls", "farres", "fqrre", "roca", "salto", "esteva", "marcade", "cardus", "anglada", "astrade", "parera", "juntanet", "pañella", "tovella", "esparchy", "garriga", "cartro", "calafi", "galofre", "nicolau", "rumeu", "casanovas", "claramunt", "pujol", "escayol", "nan", "anmatller", "alegre", "rabantos", "rafols", "gibert", "vives", "bley", "bogoña", "armengol", "jarrey", "ventura"]

**Consulta 2:** Per a cada padró de Sant Feliu de Llobregat (SFL), retorna l'any de padró, el número d'habitants, i la llista de cognoms. Elimina duplicats i "nan".

```

1 //Ex 2
2 MATCH (i:Individu)-[v:VIU]->(:Habitatge)
3 WHERE v.location = 'SFL' AND i.year IS NOT NULL
4 RETURN i.year AS any_padro, count(i) AS num_habitants, collect(DISTINCT i.surname) AS cognoms
5 ORDER BY any_padro

```

Amb la funció match seleccionem els nodes d'Individu que estiguin relacionats amb nodes d'Habitatge a través de la relació VIU. Amb el where agafem només aquells on la localització sigui Sant Feliu de Llobregat. Finalment retornem l'any el número d'habitats i els cognoms. Els resultats els ordenem segons l'any:

"any_padro"	"num_habitants"	"cognoms"
1833	1433	["nan", "carcereny", "canals", "dordal", "condeminas", "pahisa", "ramoneda", "sola", "basa", "basas", "camprubi", "jove", "duro", "grau", "capdevila", "roma", "moreno", "andres", "condeminas", "armengol", "valls", "creus", "faura", "caranto", "eta", "iglesias", "codinach", "botarell", "duran", "pares", "castells", "coca", "tapiol", "baptista", "andreu", "baleta", "avella", "panella", "briquets", "aloma", "magrina", "vancells", "esmarats", "casi", "negre", "arate portillon", "albarec"]
1838	287	["nan", "canameras", "gullart", "bonsoms", "sola", "pares", "comellas", "ramos", "molins", "marles", "fabregas", "ferrer", "vall", "subirana", "sala", "sagues", "martinella", "valles", "rovira", "campderros", "carcereny", "bofill"]
1839	1946	["nan", "morral", "briquets", "ilegible", "prats", "moreno", "sagues", "condeminas", "duran", "barveta", "monmany", "sararine", "rosell", "vallverdu", "parera", "camprubi", "cortes", "roca", "ramoneda", "aloma", "pivernat", "cañameras", "ros", "rros", "pahisa", "andreu", "claramun", "sarellas", "taraval", "farrer", "cuso", "serralavos", "dordal", "pares", "estape", "uxart", "porta", "berenguer", "brosa", "corron", "castro", "canals", "cornellas", "forns", "pinano", "aguada", "carles", "molina", "cliville", "traval", "riera", "janet", "bonich", "carnicer", "bastida", "dalmases", "mestres", "panella", "sur"]
1878	2745	["galtes", "amigo", "corrons", "marti", "castells", "planas", "sabat", "farran", "colet", "pahisa", "vidal", "sadurni", "k", "ll", "betes", "francas", "parera", "albareda", "ribas", "fanes", "ordal", "rius", "plana", "sune", "subirachs", "angles", "et", "olive", "llubet", "marce", "coll", "balcells", "codinachs", "molins", "barbeta", "fernandez", "palet", "teixido", "ç", "mallofre", "cuxart", "salabert", "dordal", "canadell", "busquets", "serra", "sallent", "trabal", "texido", "pages", "bat", "enech", "sauret", "llopart", "mas", "pibernati", "miret", "llanas", "pabe", "rosich", "bigas", "vigas", "faura", "padrerol", "xart", "baque", "pascual", "ribera", "equiezabal", "maymo", "aguade", "morato", "pi", "bodalles", "prats", "barba", "prune"]

**Consulta 3:** Dels padrons de Sant Feliu de Llobregat (SFLL) d'entre 1800 i 1845 (no inclosos), retorna la població, l'any del padró i la llista d'identificadors dels habitatges de cada padró. Ordena els resultats per l'any de padró.

```
1 //Ex 3
2 MATCH (i:Individu)-[v:VIU]-(h:Habitatge)
3 WHERE v.location = 'SFLL' AND i.year > 1800 AND i.year < 1845
4 RETURN i.year AS any_padro, count(i) AS poblacio, collect(DISTINCT h.id_lla) AS habitatges
5 ORDER BY any_padro
```

Aquí tornem a fer el mateix match que les dues consultes anteriors. Amb el where seleccionem la població i el rang d'anys, entre el 1800 i el 1845. Al final retornem l'any, la població i la llista d'identificadors habitatges. Els resultats que obtenim són els següents:

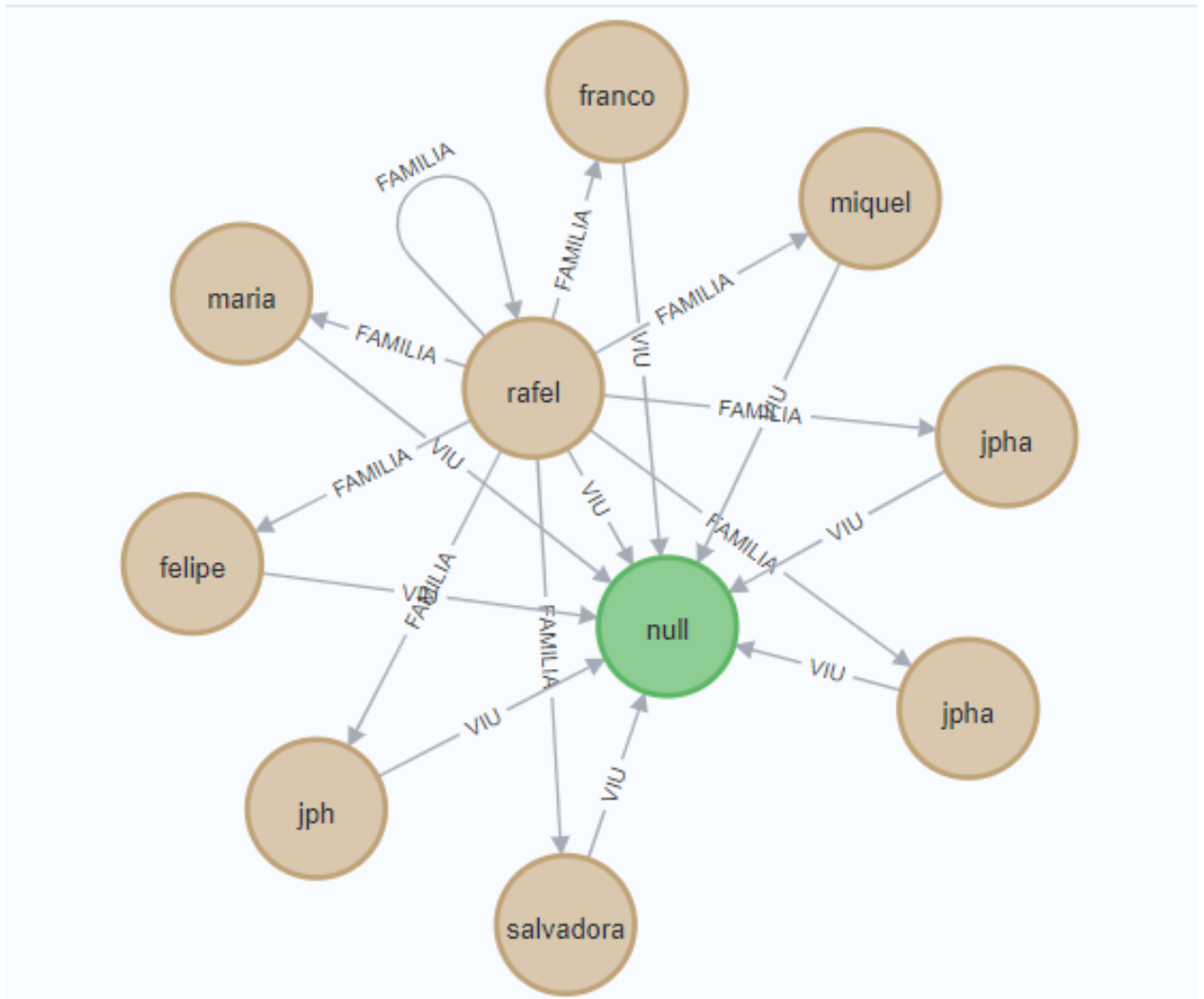
"any_padro"	"poblacio"	"habitatges"
1833	1433	[12,10,13,17,18,16,15,11,22,23,24,25,26,27,28,29,36,42,48,63,49,64,60,47,50,65,43,51,66,44,52,67,45,68,46,61,69,62,75,140,141,131,137,138,53,54,139,142,143,56,144,57,58,145,76,147,77,78,79,148,80,149,81,82,150,109,95,110,114,96,111,181,75,182,183,184,185,186,187,188,189,190,191,192,193,194,195,196,197,198,200,199,201,202,203,204,205,206,207,84,2058,264,104,274,72,74,83,88,93,94,98,99,103,106,107,108,130,105,236,238,243,244,246,247,248,250,251,252,254,255,256,292,293,294,295,296,297,298,299,300,301,302,303,304,305,306,307,308,309,310,311,312,313]
1838	287	[321,324,326,328,320,322,323,325,327,329,330,331,332,333,334,335,336,337,338,339,340,341,342,343,344,345,346,347,348]
1839	1946	[584,658,660,659,640,638,662,663,664,661,673,635,672,636,562,563,564,565,566,567,568,571,572,578,573,574,570,575,569,524,525,521,502,734,526,489,504,726,518,735,527,503,727,486,528,523,490,729,730,487,728,520,500,732,488,531,492,529,560,561,600,601,602,629,631,633,634,637,642,645,647,648,649,651,653,630,632,639,641,643,644,646,650,652,369,370,371,408,409,410,411,412,413,414,415,416,417,418,419,420,421,422,423,424,425,426,427,473,474,475,477,480,482,460,461,462,617,619,621,622,623,625,626,627,628,736,739,745,743,737,738,740,741,742,744,654,655,656,657,666,671,668,670,669,428,596,597,598,599,665,674,676,677,678,679,675,680,681,682,683,684,685,686,687,688,689,690,691,692,693,694,695,696,697]

**Consulta 4:** Retorna el nom de les persones que vivien al mateix habitatge que "rafel marti"(no té segon cognom) segons el padró de 1838 de Sant Feliu de Llobregat (SFLL). Retorna la informació en mode graf i mode llista.

```
1 //Ex 4 graph
2 MATCH (i:Individu {name: 'rafel', surname: 'marti'})-[:VIU {location: 'SFLL', year: 1838}]->
  (h:Habitatge)-[:VIU {location: 'SFLL', year: 1838}]->(other:Individu)
3 RETURN i, h, other

1 //Ex 4 llista
2 MATCH (i1:Individu {name: 'rafel', surname: 'marti'})
3 MATCH (i1)-[:VIU {location: 'SFLL', year: 1838}]->(h:Habitatge)-[:VIU {location: 'SFLL', year: 1838}]->
  (i2:Individu)
4 RETURN i1,h,i2
```

En aquesta consultes l'hem dividida en dues parts. Primer hem mostrat el graph i després amb la mateixa consulta hem canviat el return per mostrar una llista en comptes d'un graph. Ara, en el match seleccionem només l'individu que ens demanaen (Rafel Martí) i la seva localització (Sant Feliu de Llobregat), i ho relacionem amb qualsevol altre individu que també visqui en aquella localització el mateix any, 1838. A continuació veiem el graf i la llista corresponents:



"i1"	"h"	"i2"
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"surname":"nan","second_surname":"nan","name":"jpha","id":18380219}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"surname":"nan","second_surname":"nan","name":"jpha","id":18380225}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"second_surname":"nan","surname":"nan","name":"miquel","id":18380223}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"second_surname":"nan","surname":"nan","name":"franco","id":18380224}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"surname":"nan","second_surname":"nan","name":"maria","id":18380221}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"second_surname":"nan","surname":"nan","name":"felipe","id":18380222}
{"year":1838,"surname":"martí","second_surname":"nan","name":"rafel","id":18380217}	{"numero":"null","id_llar":358,"carrer":"null","any_padre":1857}	{"year":1838,"surname":"nan","second_surname":"nan","name":"jph","id":18380220}

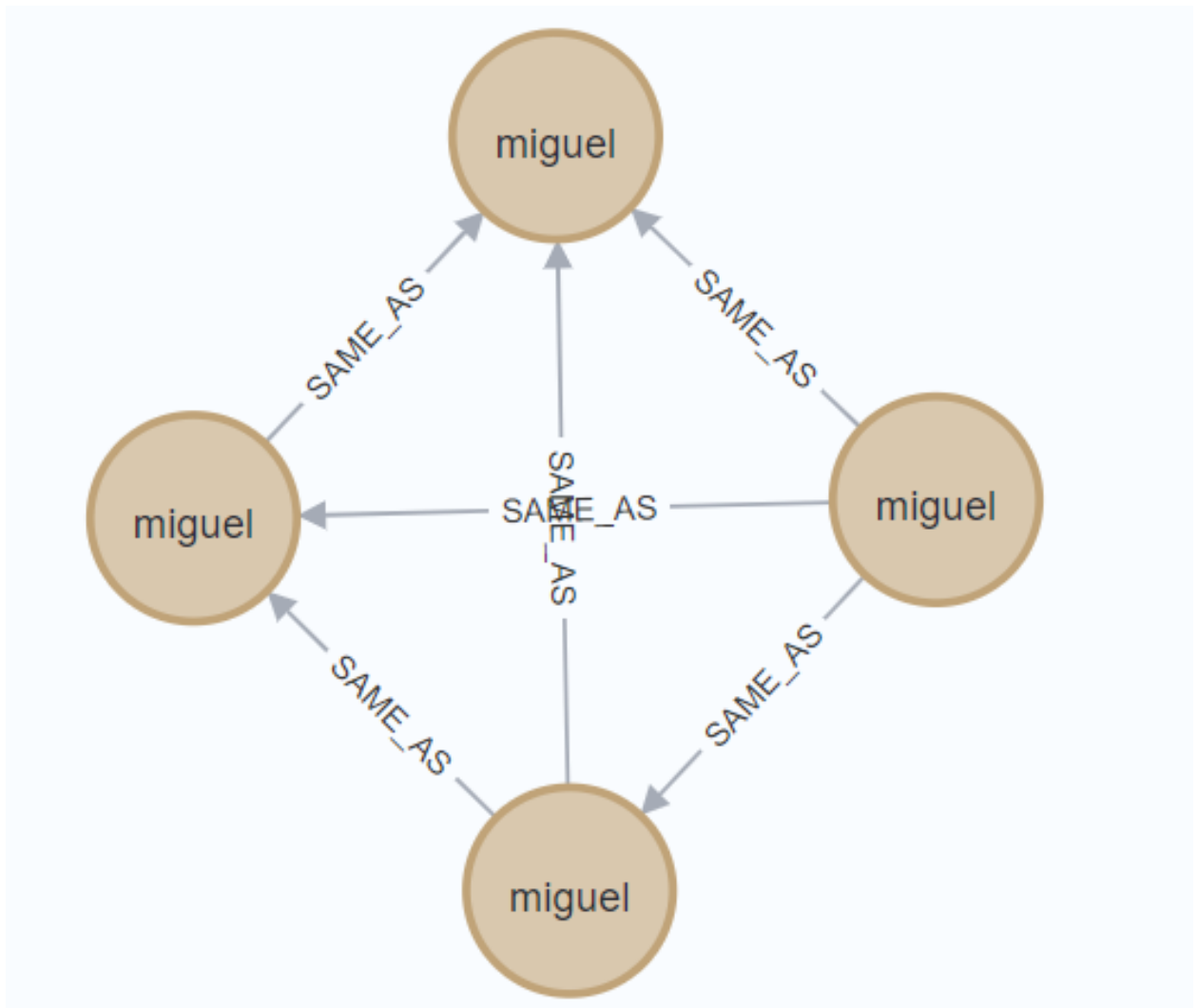
**Consulta 5:** Retorna totes les aparicions de "miguel estape bofill". Fes servir la relació SAME AS per poder retornar totes les instàncies, independentment de si hi ha variacions lèxiques (ex. diferents formes d'escriure el seu nom/cognoms). Mostra la informació en forma



de subgraf.

```
1 //Ex 5
2 MATCH (i:Individu {name: 'miguel', surname: 'estape', second_surname: 'bofill'})-[:SAME_AS*]-
  (i2:Individu)
3 RETURN i, i2
```

Aquí seleccionem l'individu Miguel Estape Bofill i amb la relació SAME\_AS seleccionem totes les aparicions. Imprimim el resultat en forma de graph:



**Consulta 6:** De la consulta anterior, retorna la informació en forma de taula: el nom, la llista de cognoms i la llista de segon cognom (elimina duplicats).

```
1 //Ex 6
2 MATCH (i:Individu {name: 'miguel', surname: 'estape', second_surname: 'bofill'})-[:SAME_AS*]-
  (i2:Individu)
3 RETURN i2.name AS nom, COLLECT(DISTINCT i2.surname) AS cognoms, COLLECT(DISTINCT i2.second_surname) AS
  segon_cognoms
```

En aquest cas, fem el mateix match que la consulta anterior. Per retornar la informació en forma de taula hem de canviar el return, de manera que mostrarem nom, cognom i segon cognom, com veiem a continuació:

"nom"	"cognoms"	"segon_cognoms"
"miguel"	["estape"]	["bufill", "bofill"]

**Consulta 7: Mostra totes les persones relacionades amb "benito julivert". Mostra la informació en forma de taula: el nom, cognom1, cognom2, i tipus de relació.**

```
1 //EX 7
2 // Mostrar totes les persones relacionades amb "Benito Julivert" en forma de taula amb el nom, cognom1, cognom2 i tipus de relació
3 MATCH (p1:Individu {name: 'benito', surname: 'julivert'})-[:FAMILIA]-(p2:Individu)
4 RETURN p2.name AS nom, p2.surname AS cognom1, p2.second_surname AS cognom2, 'FAMILIA' AS tipus_relacio
```

Primer seleccionem la persona que volem relacionar, en aquest cas, Benito Julivert i busquem totes les seves relacions. Veiem els resultats:

	nom	cognom1	cognom2	tipus_relacio
1	"jose"	"julibert"	"julia"	"FAMILIA"
2	"martin"	"julibert"	"julia"	"FAMILIA"
3	"magdalena"	"julibert"	"julia"	"FAMILIA"
4	"benito"	"julivert"	"parera"	"FAMILIA"
5	"josefa"	"julia"	"jisa"	"FAMILIA"
6	"joaquina"	"julibert"	"julia"	"FAMILIA"
7	"dolores"	"julibert"	"julia"	"FAMILIA"
8	"juan"	"julibert"	"julia"	"FAMILIA"

**Consulta 8: De la consulta anterior, mostra ara només els fills o filles de "benito julivert". Ordena els resultats alfabèticament per nom.**

```
1 //Ex 8
2 //Mostrar només els fills o filles de "Benito Julivert" i ordenar els resultats alfabèticament per nom
3 MATCH (p1:Individu {name: 'benito', surname: 'julivert'})
4 OPTIONAL MATCH (p1)-[:FAMILIA {relacio_harmonitzada: 'fill'}]-(p2:Individu)
5 OPTIONAL MATCH (p1)-[:FAMILIA {relacio_harmonitzada: 'filla'}]-(p2:Individu)
6 WHERE p2 IS NOT NULL
7 RETURN p2.name AS nom, p2.surname AS cognom1, p2.second_surname AS cognom2, 'FAMILIA' AS tipus_relacio
8 ORDER BY nom
```

En aquest cas tornem a seleccionar la mateixa persona que la consulta anterior. Per saber si és fill o filla ho fem amb la comanda OPTIONAL MATCH per a cada cas. Finalment els ordenem per nom, així se'ns ordenaran alfabèticament. Obtenim els següents resultats:

"nom"	"cognom1"	"cognom2"	"tipus_relacio"
"jose"	"julibert"	"julia"	"FAMILIA"
"juan"	"julibert"	"julia"	"FAMILIA"
"martin"	"julibert"	"julia"	"FAMILIA"

**Consulta 9:** Llisteu totes les relacions familiars que hi ha.

```
1 //EX 9
2 // Llistar totes les relacions familiars
3 MATCH ()-[r:FAMILIA]→()
4 RETURN DISTINCT r.relacio AS relacio_familiar
```

Per a poder llistar totes les relacions familiars simplement hem d'utilitzar la comanda MATCH() sobre FAMILIA. Els resultats obtinguts són els següents:

"relacio_familiar"
"null"
"hijo"
"cabeza"
"hija"
"nieto"
"yerno"
"hermano"
"nuera"
"nieta"
"madre"
"hermana"

## 5 Exercici 3

En aquest exercici analitzarem les dades del graf per entendre millor l'estructura de les dades.

### 5.1 Estudi de les components connexes (cc) i de l'estructura de les components en funció de la seva mida.

Taula agrupant els resultats segons la mida de la cc.

```
MATCH (n)
WITH n, size([(n)-[*0..3]-(node) | node]) AS ComponentSize
RETURN ComponentSize, count(*) AS NumberOfComponents
ORDER BY ComponentSize ASC
LIMIT 50
```

ComponentSize	NumberOfComponents
1	2220
2	29
3	48
4	48
5	71
6	37
7	115
8	45
9	82
10	77

Aquesta consulta retorna una taula que agrupa els resultats segons la mida de la cc (ComponentSize) i compta el nombre total de components amb aquesta mida (NumberOfComponents). Els resultats s'ordenen de forma ascendent segons la mida de la cc. Hem limitat la sortida a 50 perquè sinó ens donava problemes de memòria

Distribució de tipus de nodes (Individu o Habitatge) segons la mida de la cc.

```
MATCH (n)
WHERE NOT ()--(n)--()
WITH n
MATCH p=(n)-[*0..3]-(m)
RETURN n, collect(DISTINCT m) AS component
LIMIT 50
```