Progetto di Graph Analytics

Big Data Analytics

Corso di Laurea Magistrale in Matematica

Università di Modena e Reggio Emilia

Matteo Lombardi - 185083

1 Exploratory Data Analysis

1.1 Il dataset

Il dataset scelto per questa attività di Graph Analytics è quello presente nella sandbox di Neo4j "Crime Investigation". Il dataset è composto da dati scaricati dal sito web http://data.gov.uk/ riguardanti i crimini avvenuti nella contea inglese Great Manchester nell'agosto 2017, messi però in relazione tra di loro in modo fittizio.

Per individuare le tipologie di nodo presenti nel dataset, il loro nome e le proprietà che eventualmente possiedono e determinare il numero di ciascuna tipologia di nodo eseguo le seguenti query.

```
CALL db.schema.nodeTypeProperties()
YIELD nodeType, propertyName, propertyTypes, mandatory
RETURN nodeType, collect(propertyName) AS Properties, collect(mandatory) AS isMandatory ORDER BY nodeType
```

nodeType	Properties	isMandatory
":`Area`"	["areaCode"]	[true]
":`Crime`"	["id", "date", "type", "last_outcome", "charge", "note"]	[true, true, false, true, false, false]
":`Email`"	["email_address"]	[true]
":`Location`"	["address", "postcode", "longitude", "latitude"]	[true, true, true, true]
":`Object`"	["id", "type", "description"]	[false, true, true]
":`Officer`"	["name", "surname", "badge_no", "rank"]	[true, true, true, true]
":`Person`"	["nhs_no", "name", "surname", "age"]	[false, true, true, false]
":`PhoneCall`"	["call_date", "call_type", "call_duration", "call_time"]	[true, true, true, true]
":`Phone`"	["phoneNo"]	[true]
":`PostCode`"	["code"]	[true]
":`Vehicle`"	["make", "model", "year", "reg"]	[true, true, true, true]

```
MATCH (n)
RETURN labels(n) AS NodeType, COUNT(n) AS Count
ORDER BY NodeType
```

NodeType	Count
["Area"]	93
["Crime"]	28762
["Email"]	328
["Location"]	14904
["Object"]	7
["Officer"]	1000
["Person"]	369
["Phone"]	328
["PhoneCall"]	534
["PostCode"]	14196
["Vehicle"]	1000

Figura 1: Tipologia e numerosità dei nodi

Analogamente, per individuare le tipologie di relazione presenti nel dataset, il loro nome e le proprietà che eventualmente possiedono e determinare il numero di ciascuna tipologia di relazione eseguo le seguenti query.

```
CALL db.schema.relTypeProperties() YIELD relType, propertyName, mandatory
RETURN relType AS RelationshipType, COLLECT(propertyName) AS Properties, COLLECT(mandatory) AS IsMandatory
ORDER BY RelationshipType
```

RelationshipType	Properties	IsMandatory
":`CALLED`"	[]	[false]
":`CALLER`"	[]	[false]
":`CURRENT_ADDRESS`"	[]	[false]
":`FAMILY_REL`"	["rel_type"]	[true]
":`HAS_EMAIL`"	[]	[false]
":`HAS_PHONE`"	[]	[false]
":`HAS_POSTCODE`"	[]	[false]
":`INVESTIGATED_BY`"	[]	[false]
":`INVOLVED_IN`"	[]	[false]
":`KNOWS_LW`"	[]	[false]
":`KNOWS_PHONE`"	[]	[false]
":`KNOWS_SN`"	[]	[false]
":`KNOWS`"	[]	[false]
":`LOCATION_IN_AREA`"	[]	[false]
":`OCCURRED_AT`"	[]	[false]
":`PARTY_TO`"	[]	[false]
":`POSTCODE_IN_AREA`"	[]	[false]

```
MATCH ()-[r]→()
RETURN type(r) AS RelationshipType, COUNT(r) AS Count
ORDER BY RelationshipType
```

RelationshipType	Count
"CALLED"	534
"CALLER"	534
"CURRENT_ADDRESS"	368
"FAMILY_REL"	155
"HAS_EMAIL"	328
"HAS_PHONE"	328
"HAS_POSTCODE"	14904
"INVESTIGATED_BY"	28762
"INVOLVED_IN"	985
"KNOWS"	586
"KNOWS_LW"	80
"KNOWS_PHONE"	118
"KNOWS_SN"	241
"LOCATION_IN_AREA"	14904
"OCCURRED_AT"	28762
"PARTY_TO"	55
"POSTCODE_IN_AREA"	14196

Figura 2: Tipologia e numerosità delle relazioni

Complessivamente, posso visualizzare graficamente come le varie tipologie di relazione mettano in collegamento i vari tipi di nodo tramite la seguente query.

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

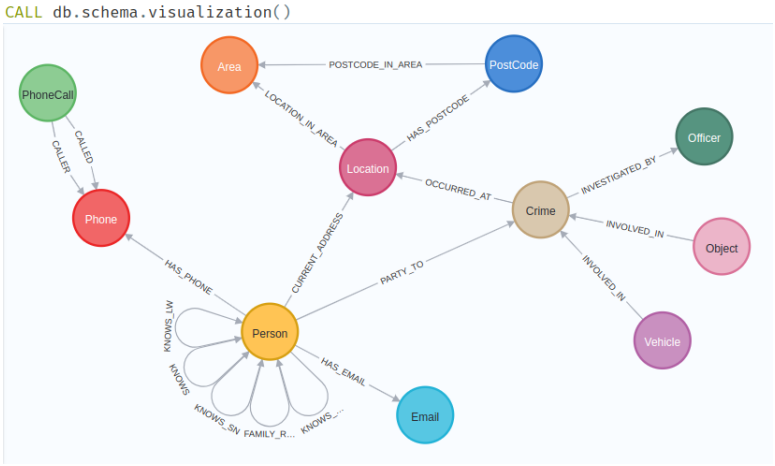


Figura 3: Tipologie di nodi e tipologie di relazioni

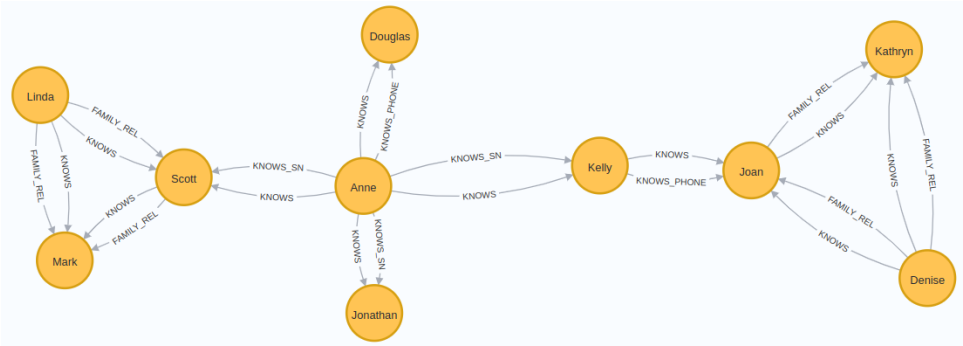
Unendo le informazioni appena ottenute, è possibile descrivere in modo più dettagliato alcuni tipi di nodo di particolare importanza.

Nodi Person

Ci sono 369 nodi di tipo **Person** collegati tra di loro tramite le relazioni:

- **FAMILY_REL**: le due persone collegate sono famigliari (genitore oppure fratello/sorella)
- **KNOWS_LW**: le due persone collegate vivono nello stesso appartamento
- **KNOWS_PHONE**: le due persone collegate si conoscono per via telefonica
- **KNOWS_SN**: le due persone collegate si conoscono sui Social Network

Come è possibile osservare nelle seguente figura, se due nodi di tipo Person sono collegati tramite una delle precedenti relazioni, allora sono collegati anche tramite la relazione **KNOWS** che esprime semplicemente che due persone, in qualche modo, si conoscono.



Inoltre, è importante osservare come gli archi siano orientati. È un aspetto da tenere in considerazione in quanto, tranne nel caso di FAMILY_REL dove viene specificato il grado di parentela, le altre relazioni sono in realtà percorribili in entrambi i sensi: se A convive con B, anche B convive con A; tuttavia, nel grafo è riportato soltanto l'arco orientato che va da A a B. Per considerare quante persone un individuo conosce in generale, ha senso, allora, concentrarsi sugli archi di tipo KNOWS e considerarli come non-orientati.

Altre tipologie di nodi con cui i nodi Person sono collegati sono:

- **Phone** tramite la relazione **HAS_PHONE**, a loro volta collegati con i nodi di tipo **PhoneCall** tramite le relazione **CALLED** e **CALLER**
- **Location** tramite la relazione **CURRENT_ADDRESS**, a loro volta collegati con i nodi di tipo **PostCode** tramite la relazione **HAS_POSTCODE**
- **Email** tramite la relazione **HAS_EMAIL**

Un'altra relazione di particolare importanza dei nodi di tipo Person è la relazione **PARTY_TO** che li collega ai nodi di tipo **Crime**: esprime la partecipazione di una persona a un reato.

Nodi Crime

Ci sono 28762 nodi di tipo **Crime**. Sono collegati, oltre che ai nodi di Person, ai nodi di tipo:

- **Location** tramite la relazione **OCCURED_AT**
- **Vehicle** tramite la relazione **INVOLVED_IN**
- **Object** tramite la relazione **INVOLVED_IN**

La relazione con i nodi di tipo Person ci permette di ottenere il numero di criminali presenti nel dataset. Tramite le seguenti query, determino il numero di criminali presenti nel dataset e osservo che due malfattori non hanno mai collaborato a uno stesso crimine.

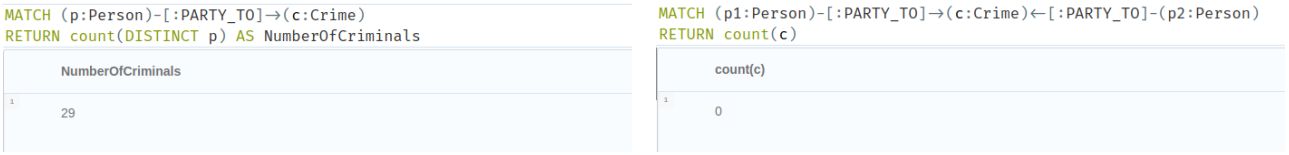


Figura 4: Numero di criminali e numero di criminali che hanno collaborato

Un'altra relazione di particolare importanza dei nodi di tipo Crime è la relazione **INVESTIGATED_BY** che li collega ai nodi di tipo **Officer**: indica l'ufficiale di polizia che ha investigato al crimine considerato.

Nodi Officer

Ci sono 1000 nodi di tipo **Officer**. Di particolare rilevanza risulta essere la proprietà **rank**. In ordine crescente di importanza, i gradi della polizia britannica presenti nel dataset sono:

- Constable
- Sergeant
- Inspector
- Chief Inspector

Nodi Area

Ci sono 93 nodi di tipo **Area** caratterizzati dalla proprietà **AreaCode**, che contiene i codici dei distretti di Great Manchester.

1.2 Exploratory Data Analysis

Mi concentro sui nodi di tipo Person e ne studio la connettività. Come già detto, ha senso concentrarsi sulle sole relazioni di tipo KNOWS e considerarle senza orientamento. Costruisco, dunque, la seguente *native projection*.

```
CALL gds.graph.project(
    'all-person',
    'Person',
    {KNOWS: {orientation: 'UNDIRECTED'}}
)
YIELD graphName, nodeCount, relationshipCount
```

	graphName	nodeCount	relationshipCount
1	"all-person"	369	1172

Figura 5: Native projection: *all-person*

Calcolo il grado dei nodi di tipo Person tramite la seguente query, calcolandone anche le statistiche di base, e ne plotto la distribuzione.

```
CALL gds.degree.stream('all-person')
YIELD nodeId, score
RETURN min(score) AS min, max(score) AS max, avg(score) AS avg, percentileCont(score, 0.5) AS median
```

min	max	avg	median
0.0	13.0	3.17615176151761 56	3.0

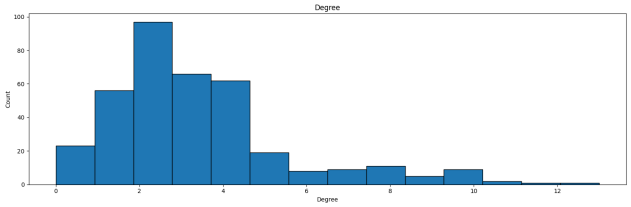


Figura 6: Degree distribution dei nodi di tipo Person

La distribuzione è asimmetrica. È possibile osservare come soltanto circa 20 persone su 369 non conoscono nessun'altro. La maggiorparte conosce da una fino a quattro persone. Poche persone hanno più di cinque conoscenze.

Per quanto riguarda la connettività globale, calcolo il diametro e il diametro effettivo considerando l'algoritmo All Pairs Shortest Paths. Mentre il primo è la distanza massima tra una qualsiasi coppia di nodi di tipo Person, il secondo è il 90° percentile di queste distanze e dà una misura più indicativa della distanza che divide due nodi qualsiasi.

```
CALL gds.allShortestPaths.stream('all-person')
YIELD distance
RETURN max(distance) AS diameter, percentileCont(distance, 0.9) AS effectiveDiameter
```

	diameter	effectiveDiameter
1	11.0	7.0

Figura 7: Diametro e diametro effettivo

Il diametro e il diametro effettivo non sono troppo diversi: il primo misura 11, il secondo misura 7. Questo permette di dedurre che le connessioni sono ben distribuite e che il sottografo dei nodi di tipo Person è particolarmente connesso.

Come ulteriore prova di ciò, individuo le componenti debolmente connesse con la seguente query.

```
CALL gds.wcc.stream('all-person')
YIELD nodeId, componentId
RETURN componentId, COUNT(nodeId) AS NumberOfNodes  ORDER BY NumberOfNodes
```

componentId	NumberOfNodes
0	342
86	2
95	2
22	1
25	1

Figura 8: Componenti debolmente connesse

Ottengo 26 componenti debolmente connesse, di cui una con 342 nodi, due con 2 nodi e tutte le altre con un solo nodo. I nodi di tipo Person si organizzano, dunque, in una zona centrale densamente connessa e in pochi nodi sparsi isolati o che conosco al più soltanto un'altra persona.

2 Prima research question

È possibile individuare delle reti criminali che ruotino attorno a determinate persone con maggiore influenza?

Per individuare delle reti criminali all'interno del dataset mi concentro sui nodi di tipo Person dove applicherò un algoritmo di community detection e calcolerò una qualche misura di centralità.

Innanzitutto, creo una *cypher projection* caricando i nodi di tipo Person che abbiano almeno un arco di tipo KNOWS che, come già osservato, esiste soltanto se due nodi sono già in collegamento tramite un arco di tipo KNOWS_SN, KNOWS_PHONE, KNOWS_LW o FAMILY_REL. Escludo, quindi, tutti i nodi di tipo Person isolati e che quindi non conoscono nessun'altra persona.

```
MATCH (source:Person)-[:KNOWS]→(target:Person)
WITH gds.graph.project(
    'person',
    source,
    target,
    {},
    {undirectedRelationshipTypes: ['*']}
) AS g
RETURN g.graphName AS Graph, g.nodeCount AS NumberOfNodes, g.relationshipCount AS NumberOfRelationships
```

	Graph	NumberOfNodes	NumberOfRelationships
1	"person"	346	1172

Figura 9: Cypher projection: *person*

Individuo, ora, delle comunità di persone utilizzando l'algoritmo di Louvain. L'algoritmo di Louvain risulta essere la scelta ottimale per questo tipo di grafo in quanto ci sono dei nodi poco connessi tra loro e, in tal caso, un metodo gerarchico, se iniziasse a determinare le comunità proprio a partire da quei nodi, non riuscirebbe a clusterizzare bene.

```
CALL gds.louvain.stream('person')
YIELD nodeId, communityId
RETURN communityId, count(*) AS NumberOfPeople, collect(gds.util.asNode(nodeId).name + ' ' +
gds.util.asNode(nodeId).surname) AS People
ORDER By NumberOfPeople DESC
```

	communityId	NumberOfPeople	People
1	267	32	["Stephanie Hughes", "Mary Young", "Bobby Russell", "Pamela Gibson", "Roger Brooks", "Maria Hughes", "Philip Shaw",
2	76	30	["Rachel Turner", "Todd Garcia", "Eric Gutierrez", "Janet Cunningham", "Ashley Robertson", "Carlos Black", "Michelle Pat
3	255	30	["Anne Nguyen", "Carlos Chavez", "Henry Jacobs", "Mildred Spencer", "Jerry Johnston", "Sharon White", "Cynthia Foster
4	283	30	["Carlos Matthews", "Thomas Harrison", "Barbara Moreno", "Carl Hayes", "Nancy Campbell", "Jeffrey Lewis", "Patricia Ca
5	185	28	["Frank Taylor", "Michael Mason", "Lillian Porter", "Craig Gordon", "Judith Moore", "Bobby Thompson", "Brian Austin", "Ha

Figura 10: Algoritmo di Louvain su *person* - modalità stream

Ho ottenuto 16 comunità, dove quella più numerosa contiene 32 persone. Di seguito la distribuzione della numerosità delle comunità.

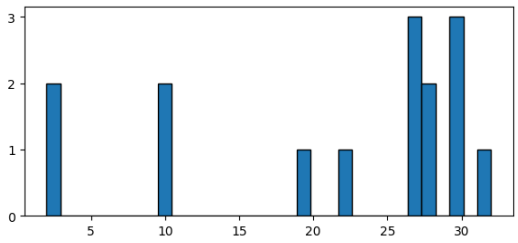


Figura 11: Numerosità comunità di *person*

Scrivo sui nodi del grafo originale la comunità a cui appartengono tramite la modalità write.

```
CALL gds.louvain.write('person', {writeProperty: 'communityId_whole'})
YIELD communityCount, nodePropertiesWritten
```

	communityCount	nodePropertiesWritten
1	16	346

Figura 12: Algoritmo di Louvain su *person* - modalità write

È utile creare una nuova proprietà *isCriminal* per i nodi di tipo Person che contenga un valore booleano a seconda che la persona in questione abbia commesso almeno un crimine. Questo è possibile tramite la seguente query.

```
MATCH (p:Person)
SET p.isCriminal =
CASE
    WHEN EXISTS((p)-[:PARTY_TO]→(:Crime)) THEN true
    ELSE false
END
```

Figura 13: Aggiunta proprietà *isCriminal* ai nodi di tipo Person

Determino, ora, le persone più influenti all'interno del grafo. Uso l'algoritmo di Pagerank che mi permette di sfruttare l'autorevolezza dei nodi per determinare l'importanza delle persone stesse all'interno del dataset.

```
CALL gds.pageRank.stream('person')
YIELD nodeId, score
RETURN collect(gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname) AS Person,
    score AS Pagerank_whole, gds.util.asNode(nodeId).isCriminal AS isCriminal
ORDER BY Pagerank_whole DESC, Person ASC
```

	Person	Pagerank_whole	isCriminal
1	["Amanda Alexander"]	3.2734100085266578	false
2	["Annie Duncan"]	2.9060462979179458	false
3	["Bruce Baker"]	2.8328271713130477	false
4	["Andrew Foster"]	2.8183272923561176	false
5	["Anne Rice"]	2.808077448047087	false

Figura 14: Pagerank su *person* - modalità stream

Osservo che tra le prime cinque persone con Pagerank più alto, nessuna è un criminale.

Scrivo sui nodi del grafo originale il punteggio Pagerank ottenuto tramite la modalità write.

```
CALL gds.pageRank.write('person', {writeProperty: 'Pagerank_whole'})
YIELD nodePropertiesWritten
```

	nodePropertiesWritten
1	346

Figura 15: Pagerank su *person* - modalità write

Mi concentro sulla comunità più grande individuata, ovvero quella con *communityId_whole = 267*. Da una delle query precedenti so che ci sono 32 persone.
Quante di queste hanno un Pagerank elevato? Quante sono criminali? Le persone con Pagerank elevato sono criminali?
Eseguo la seguente query.

```
MATCH (p:Person)
WHERE p.communityId_whole = 267
RETURN collect(p.name + ' ' + p.surname) AS Person, p.Pagerank_whole AS Pagerank, p.isCriminal AS isCriminal
ORDER BY Pagerank DESC
```

	Person	Pagerank	isCriminal
1	["Andrea George"]	2.6032183559188042	false
2	["Bonnie Gilbert"]	2.3730965388684484	false
3	["Amanda Cooper"]	1.8157897878839138	false
4	["Lillian Martinez"]	1.3527085793767764	true
5	["Eric Berry"]	1.0631332294714992	false

Figura 16: Criminali e persone con Pagerank elevato nella comunità più grande

Calcolo, inoltre, le statistiche di base del Pagerank.

```
MATCH (p:Person)
WITH p.Pagerank_whole AS pagerank
RETURN min(pagerank) AS min, max(pagerank) AS max, avg(pagerank) AS avg, percentilecont(pagerank, 0.5) AS median
```

	min	max	avg	median
1	0.3471661215576877	3.2734100085266578	0.9612404689154855	0.8393569024690324

Figura 17: Statistiche di base del Pagerank dei nodi di tipo Person

È possibile osservare che esattamente la metà dei membri di questa comunità (16 su 32 persone totali) ha un Pagerank maggiore della mediana del Pagerank di tutti i nodi di tipo Person. In particolare, soltanto i primi quattro membri hanno un Pagerank elevato rispetto al resto della comunità. Uno tra questi è un criminale. Nello specifico, i criminali presenti nella comunità più grande e il loro rispettivo Pagerank è ottenibile tramite la query in Fig. 18.

Utilizzo adesso l'algoritmo di Dijkstra per determinare il cammino minimo tra i due nodi con Pagerank maggiore (non criminali) all'interno della comunità considerata. Passa per un criminale?
Come è possibile osservare in Fig. 19, il cammino minimo tra *Andrea George* e *Bonnie Gilbert*, le due persone della comunità più grande con Pagerank maggiore, passa effettivamente per il criminale *Billy Moore*. Questo potrebbe suggerire che i rapporti e le interazioni tra queste due persone possano essere volte, per lo più, all'organizzazione di un crimine.

Analogamente, utilizzando sempre l'algoritmo di Dijkstra, determino il cammino minimo tra due criminali. Passa per una persona con Pagerank elevato?

```
MATCH (p:Person)
WHERE p.communityId_whole = 267 AND p.isCriminal = true
RETURN collect(p.name + ' ' + p.surname) AS Criminal, p.Pagerank_whole AS Pagerank
ORDER BY Pagerank DESC, Criminal ASC
```

	Criminal	Pagerank
1	["Lillian Martinez"]	1.3527085793767764
2	["Billy Moore"]	1.0379280445106063
3	["Stephanie Hughes"]	1.032455254901531
4	["Maria Hughes"]	0.5972643423264079
5	["Fred Williamson"]	0.37243552368698774

Figura 18: Criminali e relativo Pagerank della comunità più grande

```
MATCH (source:Person {name: 'Andrea', surname: 'George'}), (target:Person {name: 'Bonnie', surname: 'Gilbert'})
CALL gds.shortestPath.dijkstra.stream('person', {
    sourceNode: source,
    targetNode: target
})
YIELD index, sourceNode, targetNode, totalCost, nodeIds, costs, path
RETURN
    collect(gds.util.asNode(sourceNode).name + ' ' + gds.util.asNode(sourceNode).surname) AS sourceNodeName,
    collect(gds.util.asNode(targetNode).name + ' ' + gds.util.asNode(targetNode).surname) AS targetNodeName,
    totalCost,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname] AS nodeNames
```

	sourceNodeName	targetNodeName	totalCost	nodeNames
1	["Andrea George"]	["Bonnie Gilbert"]	3.0	["Andrea George", "Virginia Allen", "Billy Moore", "Bonnie Gilbert"]

Figura 19: Shortest Path tra le due persone con Pagerank maggiore della comunità più grande

```
MATCH (source:Person), (target:Person)
WHERE source.isCriminal=true AND target.isCriminal=true AND elementId(source)<elementId(target) AND source.communityId_whole=267 AND target.communityId_whole=267
CALL gds.shortestPath.dijkstra.stream('person', {
    sourceNode: source,
    targetNode: target
})
YIELD index, sourceNode, targetNode, totalCost, nodeIds, costs, path
RETURN
    collect(gds.util.asNode(sourceNode).name + ' ' + gds.util.asNode(sourceNode).surname) AS sourceNodeName,
    collect(gds.util.asNode(targetNode).name + ' ' + gds.util.asNode(targetNode).surname) AS targetNodeName,
    totalCost,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname] AS nodeNames
```

	sourceNodeName	targetNodeName	totalCost	nodeNames
1	["Stephanie Hughes"]	["Fred Williamson"]	2.0	["Stephanie Hughes", "Bonnie Gilbert", "Fred Williamson"]
2	["Stephanie Hughes"]	["Billy Moore"]	2.0	["Stephanie Hughes", "Bonnie Gilbert", "Billy Moore"]
3	["Maria Hughes"]	["Stephanie Hughes"]	2.0	["Maria Hughes", "Bonnie Gilbert", "Stephanie Hughes"]
4	["Maria Hughes"]	["Fred Williamson"]	2.0	["Maria Hughes", "Bonnie Gilbert", "Fred Williamson"]
5	["Maria Hughes"]	["Billy Moore"]	2.0	["Maria Hughes", "Bonnie Gilbert", "Billy Moore"]
6	["Maria Hughes"]	["Lillian Martinez"]	3.0	["Maria Hughes", "Bonnie Gilbert", "Billy Moore", "Lillian Martinez"]
7	["Fred Williamson"]	["Billy Moore"]	2.0	["Fred Williamson", "Bonnie Gilbert", "Billy Moore"]
8	["Lillian Martinez"]	["Stephanie Hughes"]	3.0	["Lillian Martinez", "Billy Moore", "Bonnie Gilbert", "Stephanie Hughes"]
9	["Lillian Martinez"]	["Fred Williamson"]	3.0	["Lillian Martinez", "Billy Moore", "Bonnie Gilbert", "Fred Williamson"]
10	["Lillian Martinez"]	["Billy Moore"]	1.0	["Lillian Martinez", "Billy Moore"]

Figura 20: Shortest paths tra le coppie di criminali della comunità più grande

Come è possibile osservare in Fig. 20, il cammino minimo tra tutte le coppie di criminali passa effettivamente per *Bonnie Gilbert* che è la persona con Pagerank maggiore all'interno della comunità considerata ed, eventualmente, per il criminale *Billy Moore*. Questo potrebbe suggerire che, anche se due criminali non partecipano mai allo stesso reato, i crimini da loro commessi vengano orchestrati da una banda alla quale appartengono persone molto influenti con Pagerank elevato, come appunto *Bonnie Gilbert*.

Questo è in linea col fatto che *Bonnie Gilbert* è uno tra i nodi di tipo Person della comunità considerata con misura di Betweenness maggiore, come è possibile osservare dalle seguenti query.

```
CALL gds.betweenness.write('person', { writeProperty: 'betweenness' })
YIELD centralityDistribution, nodePropertiesWritten
RETURN centralityDistribution.min AS minimumScore, centralityDistribution.mean AS meanScore, nodePropertiesWritten
```

	minimumScore	meanScore	nodePropertiesWritten
1	0.0	666.5664630515038	346

```
MATCH (p:Person)
WHERE p.communityId_whole=267
RETURN  p.name AS Name, p.surname AS Surname, p.betweenness AS Betweenness
ORDER BY Betweenness DESC
```

	Name	Surname	Betweenness
1	"Andrea"	"George"	3218.9709448789954
2	"Bonnie"	"Gilbert"	2408.535556440949
3	"Amanda"	"Cooper"	1228.912608633197
4	"Lillian"	"Martinez"	1050.4038644074546

Figura 21: Betweenness Centrality della comunità più grande

Questo suggerisce che quanto fatto finora potrebbe effettivamente permettere di determinare delle organizzazioni criminali potenzialmente orchestrate da persone molto influenti. Eseguo la seguente query per determinare il tipo di crimine maggiormente connesso dai criminali della comunità considerata.

```
MATCH (p:Person)-[:PARTY_TO]-(c:Crime)
WHERE p.communityId_whole=267
RETURN c.type AS Crime, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Crime	NumberOfCrimes
1	"Violence and sexual offences"	2
2	"Vehicle crime"	2
3	"Criminal damage and arson"	1
4	"Burglary"	1
5	"Robbery"	1

Figura 22: Crimini maggiormente commessi dalla comunità più grande

Osservo che non c'è una tipologia di crimine che sia stata maggiormente commessa dai criminali appartenenti alla comunità considerata.

Può essere sensato e interessante, a questo punto, concentrarsi soltanto sui criminali e sulle loro immediate conoscenze, ovvero su tutte quelle persone che sono collegate ai criminali tramite un solo arco KNOWS, e ripetere tutte quante le query appena eseguite.
Creo, allora, una *cypher projection* caricando i nodi di tipo Person che abbiano commesso almeno un crimine e tutti i loro *amici stretti*.

```
MATCH (source)-[:KNOWS]-(target)
WHERE (source.isCriminal=false AND target.isCriminal=true) OR (source.isCriminal=true AND target.isCriminal=true) AND elementId(source)<elementId(target)
WITH source, target
WITH gds.graph.project(
    'friends-criminal',
    source,
    target,
    {},
    {undirectedRelationshipTypes: ['*']}
) AS g
RETURN g.graphName AS Graph, g.nodeCount AS NumberOfNodes, g.relationshipCount AS NumberOfRelationships
```

	Graph	NumberOfNodes	NumberOfRelationships
1	"friends-criminal"	101	224

Figura 23: Cypher projection: *friends-criminal*

Individuo delle comunità di persone, in particolare di criminali e loro conoscenze, tramite l'algoritmo di Louvain.

```
CALL gds.louvain.stream('friends-criminal')
YIELD nodeId, communityId
RETURN communityId, count(*) AS NumberOfPeople, collect(gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname) AS People
ORDER By NumberOfPeople DESC
```

	communityId	NumberOfPeople	People
1	60	18	["Kelly Peterson", "Phillip Williamson", "Raymond Walker", "Kathleen Peters", "Diana Murray", "Jessica Kelly", "Kathy Wheeler", "Melissa Warren", "Alan Ward", "Brian Morales", "Philip Gardner", "Charles Alexander"
2	26	14	["Ashley Robertson", "Matthew Phillips", "Annie George", "Rachel Hunter", "Carl Lawrence", "Lois Larson", "Justin Payne", "Virginia Gibson", "Rebecca Lee", "Nicholas Mason", "Rose Parker", "Michelle Patterson", "
3	28	13	["Philip Scott", "Rebecca Long", "Linda Baker", "Evelyn Wood", "Anne Freeman", "Melissa Mills", "Norma Payne", "Sean Myers", "Ernest Clark", "Matthew Howell", "Patricia Butler", "Victor Harper", "Michael Martin"]
4	38	12	["Bonnie Gilbert", "Carlos Matthews", "Maria Hughes", "Fred Williamson", "Anna Chapman", "Kenneth Carroll", "Diane Bradley", "Jennifer Murray", "Kathryn Allen", "Joan Flores", "Denise Brown", "Kelly Robertson"]
5	47	11	["Pamela Gibson", "Stephanie Hughes", "Mary Young", "Bobby Russell", "Brandon Martin", "Amy Bailey", "Jose Green", "Ernest Thompson", "Dennis Bradley", "Raymond Williamson", "Wanda Weaver"]
6	78	11	["Christopher Patterson", "Donald Robinson", "Henry Jacobs", "Andrea Montgomery", "Ryan Smith", "Theresa Powell", "Andrea George", "Carl Fuller", "Harry Lopez", "Linda Boyd", "Rose Crawford"]
7	66	9	["Sandra Ruiz", "David Mills", "Melissa Gibson", "Annie Duncan", "Amanda Robertson", "Howard Day", "Dennis Mcdonald", "Mary Murray", "James Hudson"]
8	87	7	["Catherine White", "Carlos Black", "Louis Richards", "Walter James", "Andrea Moreno", "Craig Marshall", "Paul Arnold"]
9	73	6	["Roger Brooks", "Billy Moore", "Lillian Martinez", "Virginia Allen", "Janet Cunningham", "Jennifer Rogers"]

Figura 24: Algoritmo di Louvain su *friends-criminal* - modalità stream

Ottengo 9 comunità, di cui la più numerosa è composta da 18 persone. Scrivo sui nodi del grafo originale la comunità a cui appartengono tramite la modalità write come proprietà *communityId_friends*.

Determino ora le persone più influenti di questa proiezione. Uso l'algoritmo di Pagerank. Come è possibile osservare in Fig. 25, tra le prime sei persone con Pagerank più alto, quattro sono criminali. Scrivo sui nodi del grafo originale il punteggio Pagerank ottenuto tramite la modalità write come proprietà *Pagerank_friends*.

Mi concentro sulla comunità più grande, ovvero quella con *communityId_friends* = *60*, che contiene 18 persone.
Eseguo la query in Fig. 26 e calcolo le statistiche di base del Pagerank come in Fig. 27.

```
CALL gds.pageRank.stream('friends-criminal')
YIELD nodeId, score
RETURN collect(gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname) AS Person, score AS Pagerank_friends,
    gds.util.asNode(nodeId).isCriminal AS isCriminal
ORDER BY Pagerank_friends DESC, Person ASC
```

	Person	Pagerank_friends	isCriminal
1	["Annie George"]	4.1491976187671105	true
2	["Andrea Montgomery"]	3.957275580794182	true
3	["Amy Bailey"]	3.4556238271999984	true
4	["Amanda Robertson"]	3.2405826175190313	true
5	["Anne Freeman"]	3.1139734835956725	false
6	["Bonnie Gilbert"]	2.7779381019555127	false

Figura 25: Pagerank su *friends-criminal* - modalità stream

```
MATCH (p:Person)
WHERE p.communityId_friends = 60
RETURN p.name + ' ' + p.surname AS Person, p.Pagerank_friends AS Pagerank, p.isCriminal AS isCriminal
ORDER BY Pagerank DESC
```

	Person	Pagerank	isCriminal
1	"Alan Ward"	2.582533680788214	true
2	"Jessica Kelly"	1.9419370435405834	true
3	"Brian Morales"	1.9301406578459854	true
4	"Phillip Williamson"	1.5069638593685641	true
5	"Diana Murray"	1.3188324084810459	true

Figura 26: Criminali e amici stretti con Pagerank elevato nella comunità più grande

```
MATCH (p:Person)
WITH p.Pagerank_friends AS pagerank
RETURN min(pagerank) AS min, max(pagerank) as max, avg(pagerank) AS avg, percentilecont(pagerank, 0.5) AS median
```

	min	max	avg	median
1	0.4211497330515807	4.1491976187671105	0.9612404689154856	0.5418651071143014

Figura 27: Statistiche di base del Pagerank dei nodi di *friends-criminal*

Osservo che nella comunità considerata, più della metà delle persone (10 su 18 persone totali) ha un Pagerank maggiore della mediana del Pagerank dei nodi presenti nella proiezione *friends-criminal*. In particolare, soltanto i primi tre hanno un Pagerank elevato rispetto al resto della comunità. In ogni caso, le prime sette persone con Pagerank maggiore sono tutti criminali. C'è un ottavo criminale che risulta essere la decima persona con Pagerank più alto.

A questo punto, anche per la costruzione stessa della proiezione, è chiaro che un cammino minimo tra due criminali passerà per un'altra persona con Pagerank elevato (che in questo caso è per lo più un criminale) e, analogamente, il cammino minimo tra due persone non criminali con Pagerank elevato passerà per un criminale, come è possibile osservare, rispettivamente, nelle query in Fig. 28 e in Fig. 29.

In ultima analisi, con la query in Fig. 30 determino il tipo di crimine maggiormente commesso dai criminali della comunità considerata.

Posso concludere che procedendo in questo modo sia effettivamente possibile determinare delle reti criminali che ruotino attorno a delle persone molto influenti. Nel caso della comunità più grande, è stato possibile determinare una banda criminale esperta in furti d'auto e spaccio di droga.

```
MATCH (source:Person), (target:Person)
WHERE source.communityId_friends=60 AND target.communityId_friends=60 AND source.isCriminal=true AND target.isCriminal=true
    AND elementId(source)<elementId(target)
CALL gds.shortestPath.dijkstra.stream('friends-criminal', {sourceNode: source, targetNode: target})
YIELD sourceNode, targetNode, totalCost, nodeIds
RETURN
    collect(gds.util.asNode(sourceNode).name + ' ' + gds.util.asNode(sourceNode).surname) AS sourceNodeName,
    collect(gds.util.asNode(targetNode).name + ' ' + gds.util.asNode(targetNode).surname) AS targetNodeName,
    totalCost,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname] AS nodeNames,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).isCriminal] AS isCriminal
```

	sourceNodeName	targetNodeName	totalCost	nodeNames	isCriminal
1	["Raymond Walker"]	["Kathleen Peters"]	1.0	["Raymond Walker", "Kathleen Peters"]	[true, true]
2	["Raymond Walker"]	["Diana Murray"]	2.0	["Raymond Walker", "Kathleen Peters", "Diana Murray"]	[true, true, true]
3	["Raymond Walker"]	["Alan Ward"]	2.0	["Raymond Walker", "Phillip Williamson", "Alan Ward"]	[true, true, true]
4	["Raymond Walker"]	["Jack Powell"]	3.0	["Raymond Walker", "Phillip Williamson", "Brian Morales", "Jack Powell"]	[true, true, true, true]

Figura 28: Shortest paths tra le coppie di criminali della comunità più grande

```
MATCH (source:Person), (target:Person)
WHERE (source.name='Kathy' AND source.surname='Wheeler' AND target.name='Kelly' AND target.surname='Peterson')
    OR (source.name='William' AND source.surname='Dixon' AND target.name='Melissa' AND target.surname='Warren')
CALL gds.shortestPath.dijkstra.stream('friends-criminal', {sourceNode: source, targetNode: target})
YIELD sourceNode, targetNode, totalCost, nodeIds
RETURN
    collect(gds.util.asNode(sourceNode).name + ' ' + gds.util.asNode(sourceNode).surname) AS sourceNodeName,
    collect(gds.util.asNode(targetNode).name + ' ' + gds.util.asNode(targetNode).surname) AS targetNodeName,
    totalCost,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).name + ' ' + gds.util.asNode(nodeId).surname] AS nodeNames,
    [nodeId IN nodeIds | gds.util.asNode(nodeId).isCriminal] AS isCriminal
```

	sourceNodeName	targetNodeName	totalCost	nodeNames	isCriminal
1	["Kathy Wheeler"]	["Kelly Peterson"]	3.0	["Kathy Wheeler", "Alan Ward", "Brian Morales", "Kelly Peterson"]	[false, true, true, false]
2	["William Dixon"]	["Melissa Warren"]	5.0	["William Dixon", "Jack Powell", "Brian Morales", "Jessica Kelly", "Diana Murray", "Melissa Warren"]	[false, true, true, true, true, false]

Figura 29: Shortest Paths tra persone non criminali con Pagerank maggiore della comunità più grande

```
MATCH (p:Person)-[:PARTY_TO]-(c:Crime)
WHERE p.communityId_friends = 60
RETURN c.type AS Crime, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Crime	NumberOfCrimes
1	"Vehicle crime"	17
2	"Drugs"	11
3	"Robbery"	1

Figura 30: Crimini più frequenti nella comunità più grande

3 Seconda research question

La bravura di un ufficiale di polizia e la pericolosità di una zona sono correlati? I crimini indagati dal poliziotto più esperto si concentrano nella zona più pericolosa?

Intuitivamente, la bravura di un poliziotto e la pericolosità di una zona dovrebbero essere correlati. Più un ufficiale è bravo e più viene mandato in quelle zone in cui avvengono tanti crimini. Meno un ufficiale è esperto e meno gli vengono assegnati casi importanti e pericolosi.

Individuare le zone più pericolose è semplice. Basta contare i numeri di cammini tra i nodi di tipo Area e i nodi di tipo Crime.

```
MATCH p=(c:Crime)-[:OCCURRED_AT]-(:Location)-[:LOCATION_IN_AREA]-(a:Area)
RETURN count(p) AS NumberOfCrimes, a.areaCode AS AreaCode
ORDER BY NumberOfCrimes DESC
```

	NumberOfCrimes	AreaCode
1	975	"M1"
2	860	"BL1"
3	814	"BL3"
4	766	"M40"
5	699	"M9"

Figura 31: Area con maggior numero di crimini commessi

Rimane, tuttavia, il problema di come quantificare la bravura di un ufficiale di polizia. Ricordo, innanzitutto, che un nodo di tipo Officer è caratterizzato dalla proprietà *rank* e che, in ordine crescente di importanza, i gradi della polizia britannica presenti nel dataset sono:

- Constable
- Sergeant
- Inspector
- Chief Inspector

Inoltre, i nodi tipo Officer sono collegati ai nodi di tipo Crime tramite la relazione INVESTIGATED_BY. Pertanto, si potrebbe pensare di misurare la bravura di un poliziotto sulla base del numero di crimini su cui ha investigato e vedere, inoltre, se esista una relazione col suo grado.

I nodi di tipo Crime sono a loro volta collegati ai nodi di tipo Person tramite la relazione PARTY_TO. In particolare, tutti i nodi di tipo Person che sono collegati ai nodi di tipo Crime sono, di fatto, dei criminali. Quindi, si potrebbe pensare di misurare la bravura di un poliziotto anche contando il numero di criminali che ha individuato durante le sue investigazioni e vedere, anche in questo caso, se esista una relazione col suo grado.

Innazitutto, creo una *native projection* caricando i nodi di tipo Officer e Crime e gli archi di tipo INVESTIGATED_BY, come in Fig. 32.

Per determinare il grado entrante dei nodi di tipo Officer, utilizzo l'algoritmo di Degree Centrality in modalità stream come in Fig. 33. Avendo caricato nella proiezione gli archi di tipo INVESTIGATED_BY con l'orientamento originale, ovvero dai nodi di tipo Officer ai nodi di tipo Crime, è necessario eseguire l'algoritmo specificando come orientamento quello opposto. Stampo anche il grado di ciascun poliziotto.

```
CALL gds.graph.project(
    'officer-crime',
    ['Officer','Crime'],
    'INVESTIGATED_BY'
)
YIELD graphName AS Graph, nodeProjection, nodeCount AS NumberOfNodes, relationshipProjection, relationshipCount AS NumberOfRelationships
```

	Graph	nodeProjection	NumberOfNodes	relationshipProjection	NumberOfRelationships
1	"officer-crime"	{ "Crime": { "label": "Crime", "properties": { } }, "Officer": { "label": "Officer", "properties": { } } }	29762	{ "INVESTIGATED_BY": { "aggregation": "DEFAULT", "orientation": "NATURAL", "indexInverse": false, "properties": { }, "type": "INVESTIGATED_BY" } }	28762

Figura 32: Native projection: *officer-crime*

```
CALL gds.degree.stream('officer-crime', {orientation: 'REVERSE'})
YIELD nodeId, score
RETURN gds.util.asNode(nodeId).name+' '+gds.util.asNode(nodeId).surname  AS Officer, gds.util.asNode(nodeId).rank AS Rank, score AS investigatedCrimes
ORDER BY investigatedCrimes DESC, Officer DESC
```

	Officer	Rank	investigatedCrimes
1	"Madelon DeSousa"	"Sergeant"	50.0
2	"Cloe Ings"	"Police Constable"	47.0
3	"Kania Notti"	"Sergeant"	46.0
4	"Worthy Nettles"	"Inspector"	45.0
5	"Winonah Skynner"	"Inspector"	44.0

Figura 33: In-degree dei nodi di tipo Officer

Osservo fin da subito come i poliziotti che hanno lavorato a più crimini non siano necessariamente quelli di grado più elevato: nessuno dei primi cinque è un *Chief Inspector*. In effetti, l'andamento delle distribuzioni dei crimini indagati dai poliziotti di ciascun grado è più o meno lo stesso. L'unica differenza è che più alto è il rank del poliziotto, minore è il numero di poliziotti di quel rank. Questo vuol dire che i Chief Inspector sono in numero decisamente inferiore rispetto, ad esempio, ai Constable e soltanto pochi di loro hanno indagato su molti crimini.

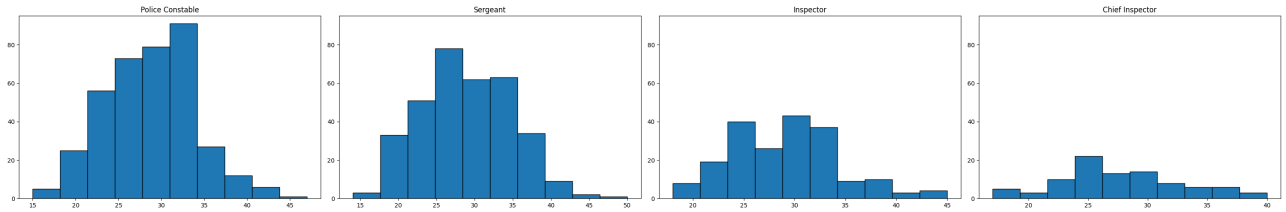


Figura 34: Distribuzione del numero di crimini indagati dai poliziotti di ciascun grado

Dunque, non c'è alcuna relazione tra il numero di crimini investigati e il grado di un ufficiale di polizia.

Proviamo a misurare la bravura degli ufficiali di polizia contando il numero di criminali da loro individuati. Eseguo la seguente query.

```
MATCH (p:Person)-[:PARTY_TO]-(:Crime)-[:INVESTIGATED_BY]-(o:Officer)
RETURN count(p) AS NumberOfCriminals, collect(p.name + ' ' + p.surname) AS Criminals, o.name + ' ' + o.surname AS Officer, o.rank AS Rank
ORDER BY NumberOfCriminals DESC, Officer ASC
```

	NumberOfCriminals	Criminals	Officer	Rank
1	3	["Raymond Walker", "Jack Powell", "Jack Powell"]	"Devy Larive"	"Police Constable"
2	1	["Phillip Williamson"]	"Ailis Crush"	"Sergeant"
3	1	["Jessica Kelly"]	"Aldric Adney"	"Police Constable"
4	1	["Annie George"]	"Aubine Stanman"	"Police Constable"

Figura 35: Numero di criminali individuati da ogni ufficiale

Si deduce che i 29 criminali presenti nel dataset hanno partecipato a dei crimini su cui hanno lavorato sempre ufficiali di polizia diversi, tranne nel caso del constable *Devy Larive* che ha individuato il criminale *Raymond Walker* e incastrato due volte *Jack Powell*.

Dal momento che il dataset è poco completo e contiene pur sempre dati fittizi, a questo punto, non avendo un vero e proprio metro per misurare la bravura di un ufficiale di polizia, considero gli ufficiali che hanno investigato su più crimini e, poi, i *Chief Inspector* che hanno investigato su più crimini. Gli ufficiali che hanno investigato su più crimini sono dati dalla seguente query.

```
MATCH (c:Crime)-[:INVESTIGATED_BY]-(o:Officer)
RETURN o.name + ' ' + o.surname AS Officer, o.rank AS Rank, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Officer	Rank	NumberOfCrimes
1	"Madelon DeSousa"	"Sergeant"	50
2	"Cloe Ings"	"Police Constable"	47
3	"Kania Notti"	"Sergeant"	46

Figura 36: Ufficiali di polizia che hanno investigato su più crimini

Tramite la query in Fig. 37 verifico che *Madelon DeSousa*, l'ufficiale di polizia che ha indagato su più casi e di grado Sergeant, abbia effettivamente lavorato maggiormente nella zona M1, che risulta essere la più pericolosa di Greater Manchester.

```
MATCH (o:Officer)-[:INVESTIGATED_BY]-(c:Crime)-[:OCCURRED_AT]-(:Location)-[:LOCATION_IN_AREA]-(a:Area)
WHERE o.name = 'Madelon' AND o.surname = 'DeSousa'
RETURN a.areaCode AS Area, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Area	NumberOfCrimes
1	"M30"	3
2	"M14"	3
3	"WN7"	2

Figura 37: Aree in cui *Madelon DeSousa* ha lavorato

Non c'è un'area in cui si sono concentrati i crimini su cui *Madelon DeSousa* ha investigato. Tuttavia, ha lavorato nell'area M14 che è la settima area di Great Manchester più pericolosa con 562 crimini.

Analogamente, individuo i *Chief Inspector* che hanno lavorato a più casi.

```
MATCH (c:Crime)-[:INVESTIGATED_BY]-(o:Officer {rank: 'Chief Inspector'})
RETURN o.name + ' ' + o.surname AS Officer, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Officer	NumberOfCrimes
1	"Kort Monelli"	40
2	"Urban Stave"	39
3	"Roberto Febry"	38

Figura 38: Chief Inspector che hanno investigato su più crimini

Tramite la seguente query verifico che *Kort Monelli* abbia effettivamente lavorato maggiormente nella zona M1.

```
MATCH (o:Officer)-[:INVESTIGATED_BY]-(c:Crime)-[:OCCURRED_AT]-(:Location)-[:LOCATION_IN_AREA]-(a:Area)
WHERE o.name = 'Kort' AND o.surname = 'Monelli'
RETURN a.areaCode AS Area, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Area	NumberOfCrimes
1	"M20"	3
2	"M32"	2
3	"SK5"	2

Figura 39: Aree in cui *Kort Monelli* ha lavorato

Anche in questo caso, non c'è un'area in cui si sono concentrati i crimini su cui *Kort Monelli* ha investigato. Tuttavia, ha lavorato nell'area M20 che è la quattordicesima (su 92) area di Great Manchester più pericolosa con 497 crimini.

Osservando la mappa in Fig. 40, i crimini su cui hanno investigato i due ufficiali considerati risultano essere avvenuti in più zone della contea di Greater Manchester, anche molto distanti tra di loro. Questo risulta essere piuttosto inverosimile, in quanto è improbabile che nello stesso mese *Madelon DeSousa* e *Kort Monelli* abbiano lavorato in così tante zone diverse; inoltre, normalmente, un poliziotto fa riferimento a una sola centrale di polizia e si limita a lavorare nelle zone limitrofe.

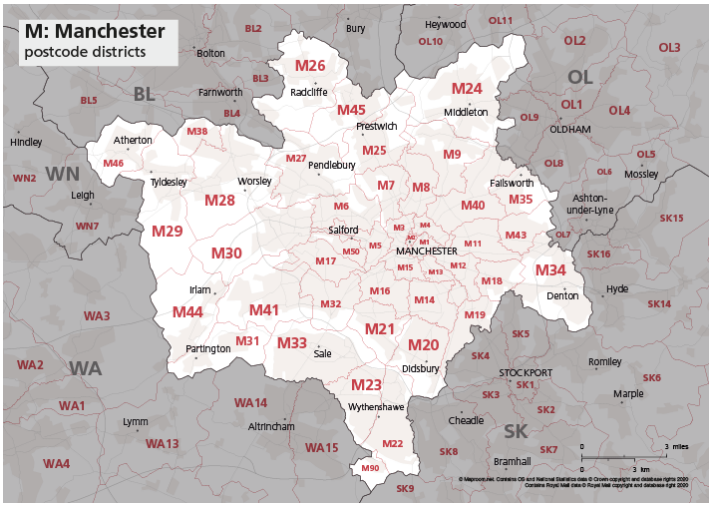


Figura 40: Distretti di Greater Manchester

Anche se provo a fare il ragionamento inverso, ovvero cercare gli ufficiali di polizia che abbiano lavorato a più casi nell'area M1, non ottengo nessun poliziotto che spicchi sugli altri in termini di numero di crimini investigati nell'area più pericolosa.

```
MATCH (o:Officer)-[:INVESTIGATED_BY]-(c:Crime)-[:OCCURRED_AT]-(:Location)-[:LOCATION_IN_AREA]-(a:Area)
WHERE a.areaCode = 'M1'
RETURN o.name + ' ' + o.surname AS Officer, o.rank AS Rank, count(c) AS NumberOfCrimes
ORDER BY NumberOfCrimes DESC
```

	Officer	Rank	NumberOfCrimes
1	"Karilynn Stanney"	"Police Constable"	4
2	"Lyle Huntingford"	"Police Constable"	4
3	"Manuel McVanamy"	"Sergeant"	4
4	"Chrotoem Rowena"	"Police Constable"	4
5	"Stewart Rintoul"	"Police Constable"	4

Figura 41: Ufficiali di polizia che hanno investigato su più crimini nell'aree M1

Questo è dovuto al fatto che i dati sono fittizi.

4 Conclusione

La prima research question aveva l'obiettivo di individuare delle comunità di malfattori che facessero riferimento a persone molto influenti, potenziali *menti* delle bande criminali. È stato possibile trovare queste reti criminali facendo Community Detection e calcolando il punteggio di Pagerank per determinare le persone più influenti. Il risultato migliore si è ottenuto restringendo il campo di ricerca ai soli criminali e alle loro immediate conoscenze, dove si è visto che la comunità più grande sia in effetti una banda criminale esperta in furti d'auto e spaccio di droga.

La seconda research question aveva l'obiettivo di verificare che gli ufficiali di polizia migliori avessero lavorato nella zona più pericolosa di Great Manchester durante il mese di Agosto 2017. Non sono stati ottenuti risultati troppo interessanti a causa dell'incompletezza ed inconsistenza del dataset: è inverosimile che in un mese siano stati commessi 28762 crimini in 93 distretti diversi della contea di Great Manchester e che 1000 ufficiali abbiano individuato soltanto 29 criminali.