Poročilo naloga Neo4j

Primož Ratej Cvahte

Podatkovne baze II

ITK-VS

Jan-2018

Vsebina

Import data

About cypher

Cypher je deklarativni grafični poizvedbeni jezik, ki omogoča ekspresivno in učinkovito poizvedovanje in posodabljanje dela nad grafi. Cypher je relativno preprost, vendar zelo močan jezik. Zelo zapletene poizvedbe v bazo podatkov lahko enostavno izrazite s Cypherjem. To omogoča uporabnikom, da se osredotočijo na svojo domeno, namesto da bi se izgubili v dostopu do baze podatkov.

Import cypher

Najprej smo uvozili podatke z poizvedovalnim jezikom cypher. Zaganjamo vsak query posebej. Podatke pa zajemamo z GitHub-a.

```
// Legis-graph LOAD CSV cypher script
// https://github.com/legis-graph/legis-graph

// Load Legislators

CREATE INDEX ON :Legislator(bioguideID);
CREATE INDEX ON :Legislator(thomasID);
CREATE INDEX ON :Legislator(lisID);
CREATE INDEX ON :Legislator(govtrackID);
CREATE INDEX ON :Legislator(opensecretsID);
CREATE INDEX ON :Legislator(votesmartID);
CREATE INDEX ON :Legislator(cspanID);
CREATE INDEX ON :Legislator(wikipediaID);
CREATE INDEX ON :Legislator(washpostID);
CREATE INDEX ON :Legislator(icpsrID);

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/legislators-
current.csv' AS line
WITH line WHERE line.thomasID IS NOT NULL
MERGE (legislator:Legislator { thomasID: line.thomasID })
```

```
    ON CREATE SET legislator = line
    ON MATCH SET legislator = line
MERGE (s:State {code: line.state})
CREATE UNIQUE (legislator)-[:REPRESENTS]->(s)
MERGE (p:Party {name: line.currentParty})
CREATE UNIQUE (legislator)-[:IS_MEMBER_OF]->(p)
MERGE (b:Body {type: line.type})
CREATE UNIQUE (legislator)-[:ELECTED_TO]->(b);

// Load Bills

USING PERIODIC COMMIT
LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/bills.csv'
AS line
MERGE (bill:Bill { billID: line.billID })
    ON CREATE SET bill = line
    ON MATCH SET bill = line;
CREATE INDEX ON :Bill(billID);

// Load

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/subjects.csv'
AS line
MERGE (subject:Subject { title: line.title });
CREATE INDEX ON :Subject(title);

// Load Congresses

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/congresses.csv'
AS line
MERGE (congress:Congress { number: line.number });

// Laod Bills Congresses

USING PERIODIC COMMIT
LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-
graph/master/outputs/bill_congresses.csv'
AS line
MATCH (bill:Bill { billID: line.billID }),
      (congress:Congress { number: line.number })
MERGE (bill)-[r:PROPOSED_DURING]->(congress);

// Load Bills Subjects

USING PERIODIC COMMIT
LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-
graph/master/outputs/bill_subjects.csv'
AS line
MATCH (bill:Bill { billID: line.billID }),
      (subject:Subject { title: line.title })
MERGE (bill)-[r:DEALS_WITH]->(subject);

// Load Bills Legislators

// Load current sponsorships
USING PERIODIC COMMIT
LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/sponsors.csv'
AS line
MATCH (bill:Bill { billID: line.billID }),
      (legislator:Legislator { bioguideID: line.bioguideID })
MERGE (bill)-[r:SPONSORED_BY]->(legislator)
    ON CREATE SET r.cosponsor = line.cosponsor;

// Load Votes

USING PERIODIC COMMIT
LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/votes.csv'
AS line
MATCH (bill:Bill { billID: line.billID }),
      (legislator:Legislator { bioguideID: line.bioguideID })
```

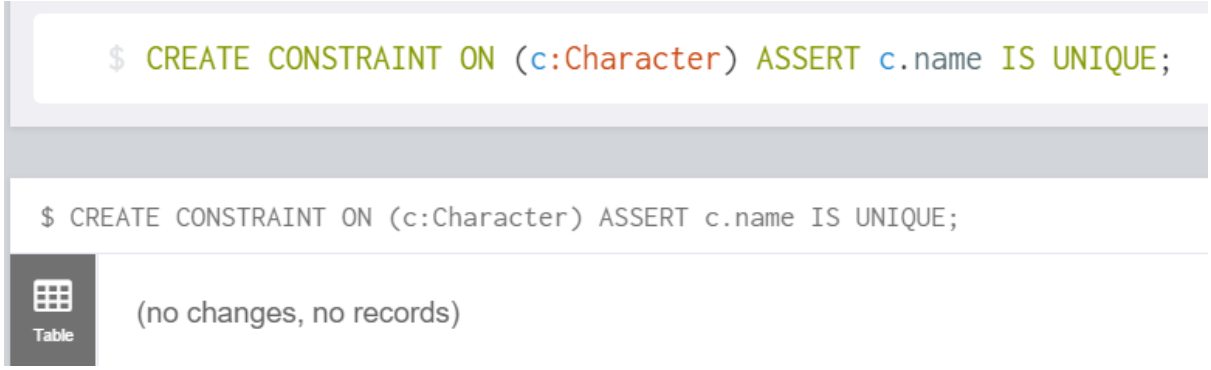
```
MERGE (bill)<-[r:VOTED_ON]-(legislator)
    ON CREATE SET r.vote = line.vote;

// Load Committees

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/committees-
current.csv' AS line
MERGE (c:Committee {thomasID: line.thomasID})
  ON CREATE SET c = line
  ON MATCH SET c = line;

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-
graph/master/outputs/bill_committees.csv' AS line
MATCH (b:Bill {billID: line.billID})
MATCH (c:Committee {thomasID: line.committeeID})
CREATE UNIQUE (b)-[:REFERRED_TO]->(c);

CREATE INDEX ON :Committee(thomasID);

// Load Committee Members

LOAD CSV WITH HEADERS
FROM 'https://raw.githubusercontent.com/legis-graph/legis-graph/master/outputs/committee-
members.csv' AS line
MATCH (c:Committee {thomasID: line.committeeID})
MATCH (l:Legislator {bioguideID: line.legislatorID})
CREATE UNIQUE (l)-[r:SERVES_ON]->(c)
SET r.rank = line.rank;

// Create District nodes
LOAD CSV WITH HEADERS
FROM 'https://github.com/legis-graph/legis-
graph/blob/master/outputs/cb_2014_districts.csv?raw=true' AS line
CREATE (d:District)
SET d.state = line.state,
    d.district = line.district,
    d.wkt = line.polygon
WITH d,line
MATCH (l:Legislator) WHERE l.state = line.state AND l.district = line.district
CREATE UNIQUE (l)-[:REPRESENTS]->(d);
```

Import CSV

Datoteka CSV Game of Thrones prikazuje povezave oz. sodelovanja med karakterji ter moč med njihovimi povezavami. Datoteko CSV prenesemo z eštudija.

Najprej moramo ustvariti omejitev, da bi potrdili celovitost naše sheme:

```
CREATE CONSTRAINT ON (c:Character) ASSERT c.name IS UNIQUE;
```

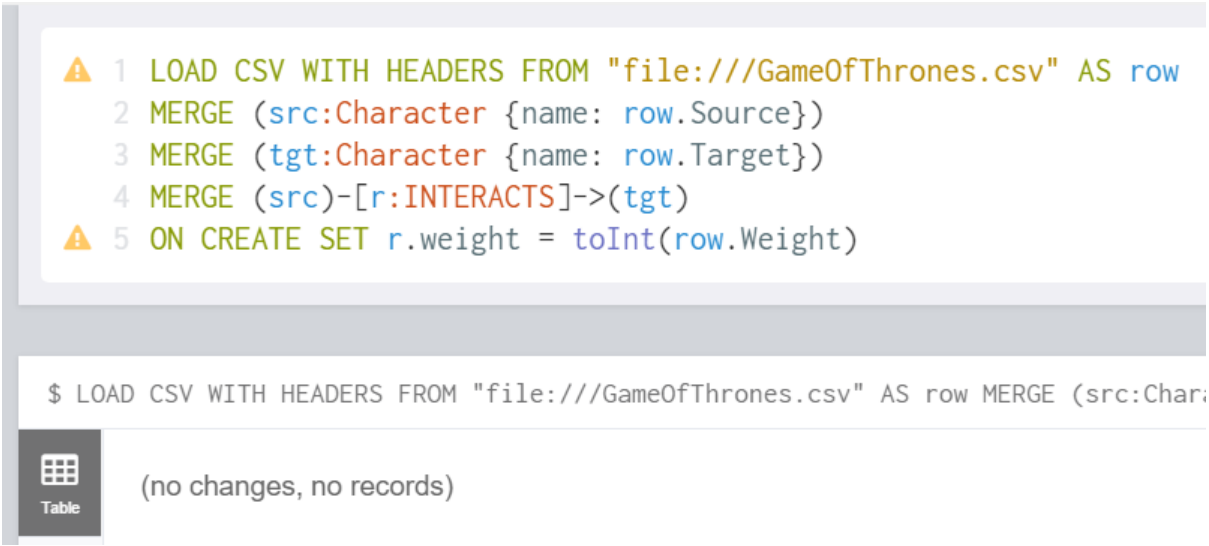


Slika 1 Create constraint

Datoteko CSV prenesemo v mapo Import ter jo uvozimo.

```
LOAD CSV WITH HEADERS FROM "file:///GameOfThrones.csv" AS row
```

```
MERGE (src:Character {name: row.Source})
MERGE (tgt:Character {name: row.Target})
MERGE (src)-[r:INTERACTS]->(tgt)
ON CREATE SET r.weight = toInt(row.Weight)
```



Slika 2 Import CSV

American Congress

1. Izpišite vse člane kongresa, ki se pišejo Smith.

```
MATCH (n {lastName:"Smith"}) return n;
```

Slika 3 Kongres_01

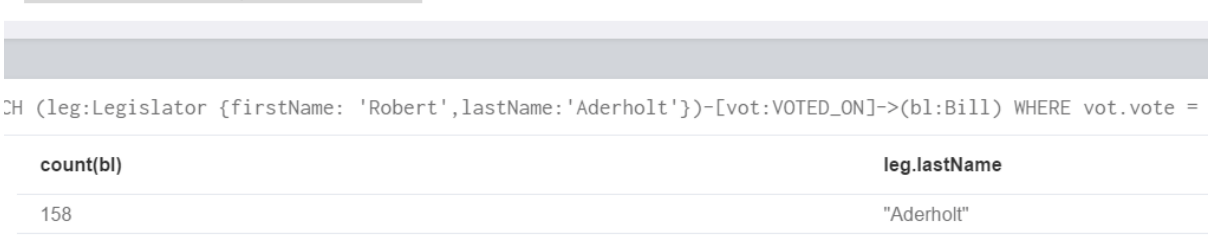
2. Izpišite vse zakone, ki so jih sponzorili republikanci iz Kalifornije, ki so starejši od 65 let.

```
MATCH (leg:Legislator), bl=(bill)-[sp_by:SPONSORED_BY]-
>(:Legislator)
WHERE leg.state="CA" AND leg.birthday>"1952-02-09" RETURN bl LIMIT
20;
```

Slika 4 Kongres_02

3. Preštejte, za koliko zakonov je pozitivno glasoval Robert Aderholt.

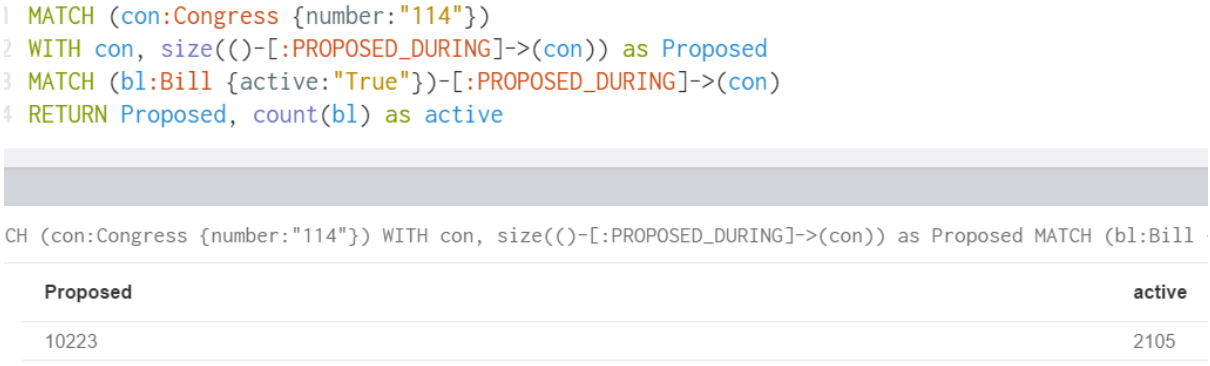
```
MATCH (leg:Legislator {firstName: 'Robert',lastName:'Aderholt'})-
[vot:VOTED_ON]->(bl:Bill)
WHERE vot.vote = 'Yea' OR vot.vote = 'Aye'
RETURN count(bl), leg.lastName
```



Slika 5 Kongres_03

4. Koliko zakonov je bilo skupno predlaganih na 114. kongresu, in koliko od teh je trenutno aktivnih?

```
MATCH (con:Congress {number:"114"})
WITH con, size(()-[:PROPOSED_DURING]->(con)) as Proposed
MATCH (bl:Bill {active:"True"})-[:PROPOSED_DURING]->(con)
RETURN Proposed, count(bl) as active
```



Slika 6 Kongres_04

5. Izpišite vse zvezne države, ki imajo v kongresu večino demokratov.

```
MATCH (leg:Legislator)-[rep:REPRESENTS]->(sta:State)
WITH sta AS drzave, sum(size((leg{currentParty:"Democrat"})-[rep]-
>(sta))) AS sumDemo,
sum(size((leg{currentParty:"Republican"})-[rep]->(sta))) AS sumRep
WHERE sumDemo>sumRep
RETURN drzave.code
```

```
MATCH (leg:Legislator)-[rep:REPRESENTS]->(sta:State)
WITH sta AS drzave, sum(size((leg{currentParty:"Democrat"})-[rep]->(sta))) AS sumDemo,
sum(size((leg{currentParty:"Republican"})-[rep]->(sta))) AS sumRep
WHERE sumDemo>sumRep
RETURN drzave.code
```

```
(leg:Legislator)-[rep:REPRESENTS]->(sta:State) WITH sta AS drzave, sum(size((leg{currentParty:"Demo
```

drzave.code
"HI"
"MA"
"RI"
"NY"
"MP"
"OR"

Slika 7 Kongres_05

6. S pomočjo grafa vizualizirajte vse predstavnike zvezne države Ohio, izvoljene v obe telesi kongresa, pri čemer prikažite tudi njihovo pripadnost stranki.

```
MATCH (sta:State {code: "OH"})<-[:REPRESENTS]-(leg:Legislator)
MATCH (leg)-[imo:IS_MEMBER_OF]->(par:Party)
WITH par.name as party, count(imo) as count
RETURN party, count;
```

```
MATCH (sta:State {code: "OH"})<-[:REPRESENTS]-(leg:Legislator)
MATCH (leg)-[imo:IS_MEMBER_OF]->(par:Party)
WITH par.name as party, count(imo) as count
RETURN party, count;
```

```
CH (sta:State {code: "OH"})<-[:REPRESENTS]-(leg:Legislator) MATCH (leg)-[im
```

party
"Democrat"
"Republican"

Slika 8 Kongres_06

7. Vizualizirajte vse senatorje iz Floride, in prikažite v katerih komitejih so aktivni.

```
MATCH (leg:Legislator)-[el_to:ELECTED_TO]->(bo:Body{type:"Senate"})
MATCH (leg)-[ser:SERVES_ON]->(com:Committee)
MATCH (leg)-[rep:REPRESENTS]->(sta:State{code:"FL"})
RETURN leg, com, bo, sta;
```

8. Kakšne zakone obravnavajo komiteji iz točke 7, in s kakšno problematiko (predmet zakona) se ti zakoni ukvarjajo?

```
MATCH (leg:Legislator)-[se:SERVES_ON]->(com:Committee)
MATCH (leg)-[rep:REPRESENTS]->(st:State{code:"FL"})
MATCH (bi:Bill)-[de_wi:DEALS_WITH]->(sub:Subject)
RETURN sub
LIMIT 20;
```

Game of thrones

1. Koliko je minimalno, maksimalno in povprečno število karakterjev, s katerimi so sodelovali karaterji?

```
MATCH (cha:Character)-[:INTERACTS]->()
WITH cha, count(*) AS num
RETURN min(num) AS min, max(num) AS max, avg(num) AS avg
```

```
MATCH (cha:Character)-[:INTERACTS]->()
WITH cha, count(*) AS num
RETURN min(num) AS min, max(num) AS max, avg(num) AS avg
```

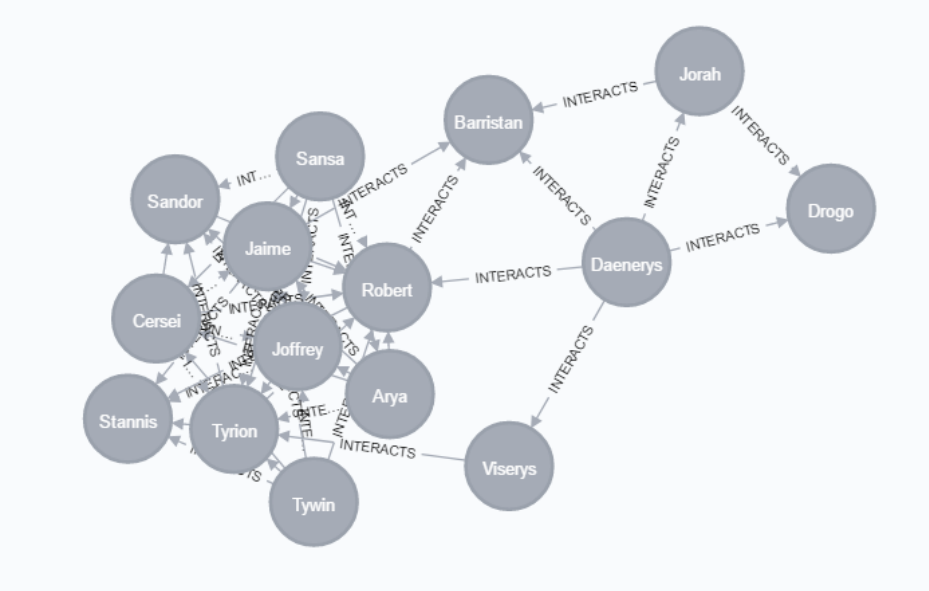
```
(cha:Character)-[:INTERACTS]->() WITH cha, count(*) AS num RETURN min(num) AS min, max(num
```

min	max	avg
1	24	4.957746478873241

Slika 9 GOT_01

2. Poiščite vse možne najkrajše poti med karakterjema Jeoffrey Lanister in Khal Drogo.

```
MATCH (dick:Character{name:"Joffrey"}),
(Khal:Character{name:"Drogo"})
MATCH path=allShortestPaths((dick)-[INTERACTS*]-(Khal))
RETURN path;
```



Slika 10 GOT_02

3. Poiščite maksimalno najkrajšo pot med dvema karakterjema.

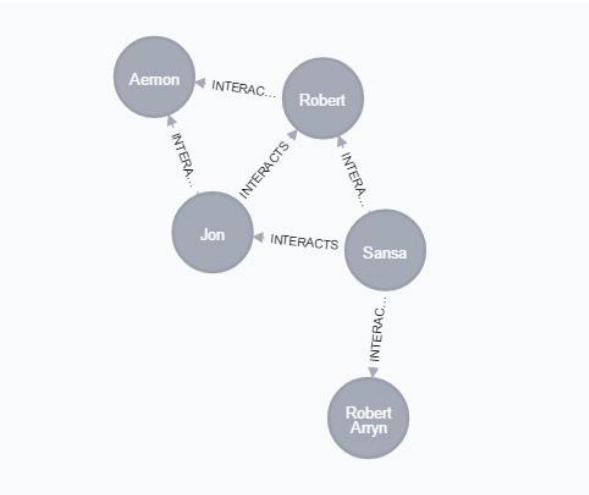
```
MATCH (char_x:Character), (char_y:Character) WHERE id(char_x) >
id(char_y)
MATCH nodes=shortestPath((char_x)-[:INTERACTS*]-(char_y))
RETURN length(nodes) AS nodes, extract(x IN nodes(nodes) | x.name)
AS numOfNodes
ORDER BY nodes DESC LIMIT 5;
```

nodes	numOfNodes
6	["Illyrio", "Belwas", "Daenerys", "Robert", "Tywin", "Oberyn", "Amory"]
6	["Illyrio", "Belwas", "Daenerys", "Robert", "Sansa", "Bran", "Luwin"]
6	["Illyrio", "Belwas", "Daenerys", "Robert", "Sansa", "Bran", "Jojen"]
6	["Nan", "Bran", "Sansa", "Robert", "Daenerys", "Belwas", "Illyrio"]
6	["Illyrio", "Belwas", "Daenerys", "Robert", "Stannis", "Davos", "Shireen"]

Slika 11 GOT_03

4. Poiščite vse karakterje, ki se pojavijo na vsaki najkrajši možni poti med karakterjema Robert Arryn in Aegon.

```
MATCH (Robert:Character {name: "Robert Arryn"}), (Aemon:Character
{name: "Aemon"})
MATCH nodes=allShortestPaths((Robert)-[INTERACTS*]-(Aemon))
RETURN nodes;
```



Slika 12 GOT_04

5. Izpišite prvih pet karakterjev, ki imajo največ povezav z drugimi karakterji, in prvih pet karakterjev, ki imajo najmanj povezav z drugimi karakterji (in izpišite število povezav).

```
MATCH (char:Character)
RETURN char.name AS Character, size( (char)-[:INTERACTS]-() ) AS min
ORDER BY min LIMIT 5
```

Character	min
"Shireen"	1
"Walton"	1
"Amory"	1
"Karl"	1
"Illyrio"	1

Slika 13 GOT_05.1

```
MATCH (char:Character)
RETURN char.name AS Character, size( (char)-[:INTERACTS]-() ) AS max
ORDER BY max DESC LIMIT 5;
```

```
CH (char:Character) RETURN char.name AS Character, size( (char)-[:INTERACTS]-() ) AS max  ORDER BY max DESC LIM
```

Character	max
"Tyrion"	36
"Sansa"	26
"Jon"	26
"Robb"	25
"Jaime"	24

Slika 14 GOT_05.2

6. Izpišite prvih pet (in zadnjih pet) karakterjev, ki imajo največjo povprečno težo sodelovanj z drugimi karakterji.

```
MATCH (char:Character)-[inter:INTERACTS]->()
RETURN DISTINCT(char.name) AS character, AVG(inter.weight) AS weight
ORDER BY weight LIMIT 5;
```

character	weight
"Balon"	4
"Luwin"	4
"Varys"	4
"Podrick"	4
"Robert Arryn"	4

Slika 15 GOT_06.1

```
MATCH (char:Character)-[inter:INTERACTS]->()
RETURN DISTINCT(char.name) AS character, AVG(inter.weight) AS weight
ORDER BY weight DESC LIMIT 5;
```

character	weight
"Hodor"	38
"Bran"	31.444444444444443
"Jojen"	22
"Samwell"	19.000000000000004
"Jon"	18.904761904761905

Slika 16 GOT_06.1