UNIVERZITET U NOVOM SADU

FAKULTET TEHNIČKIH NAUKA

Milena Kovačević

Neo4j graf baza podataka

SEMINARSKI RAD

- Master akademske studije –

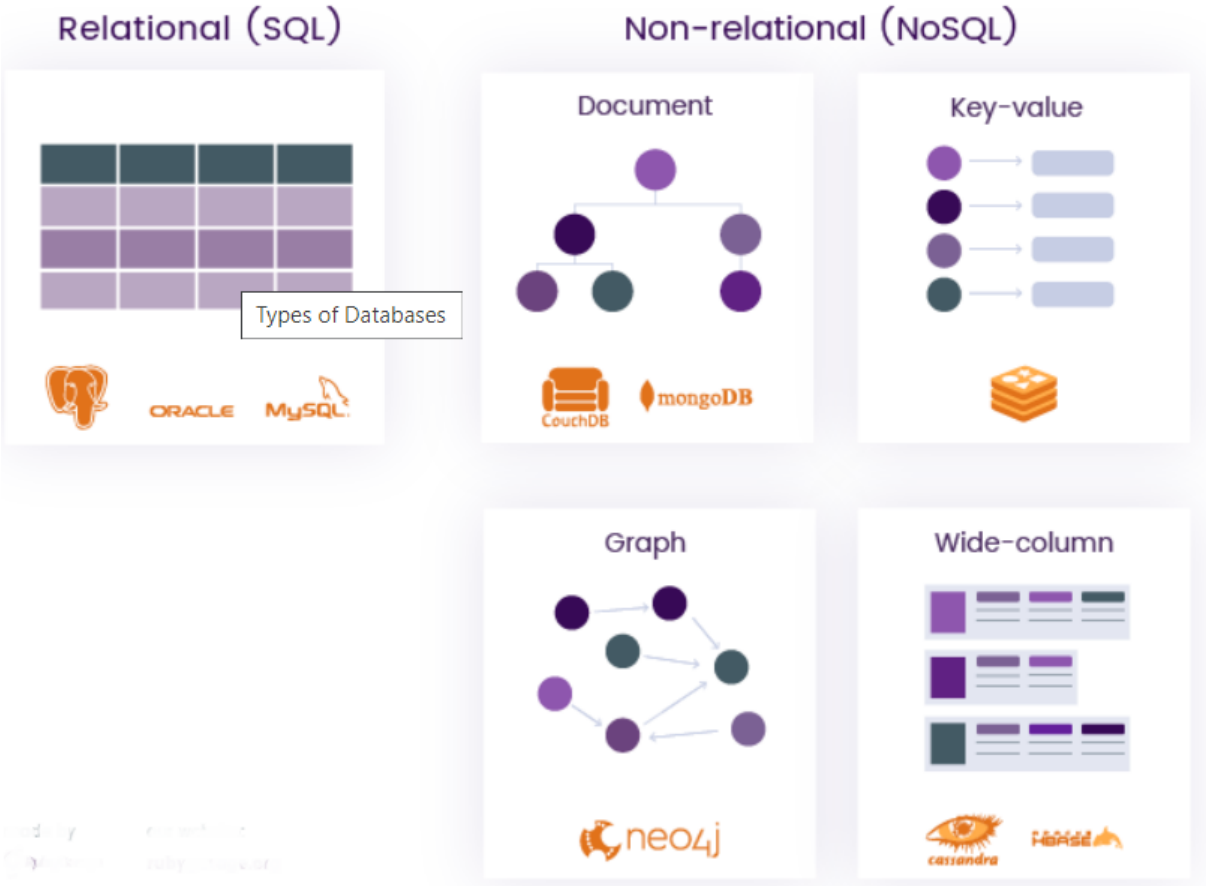
Novi Sad, 2022

Sadržaj

NoSQL graf baza podataka

Sistemi pružaju ogromnu količinu podataka o svemu, pritom često bez definisanog načina pomoću kojeg će se analizirati, skladištiti ili upravljati tim podacima, pa je samim tim veoma nezahvalno pokušati skladištiti taj ogroman skup podatka, bez određene strukture, u tradicionalni SQL model. Odatle se i izrodio “Not Only SQL”, poznat kao NoSQL.

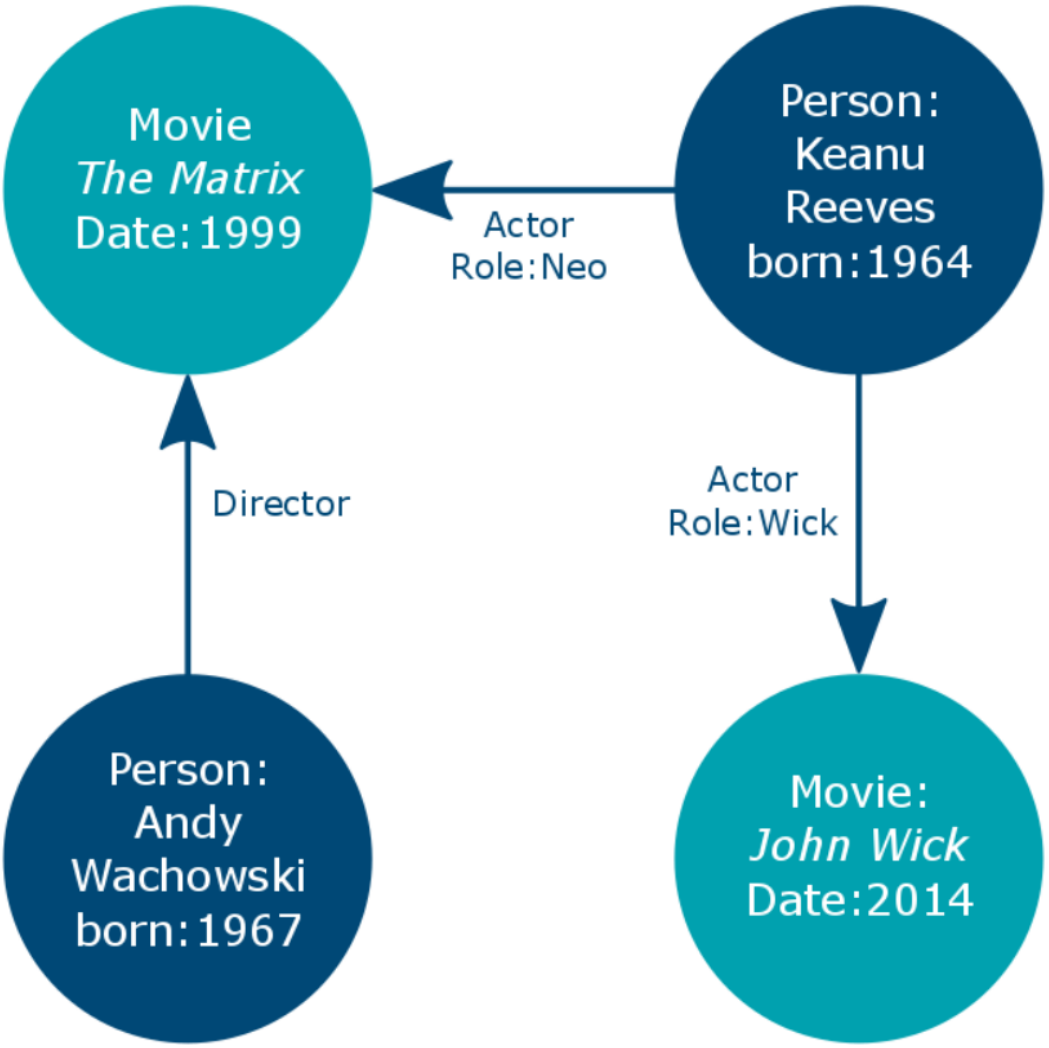
Unutar NoSQL-a imamo izbor različitiih struktura podataka (Slika 1.1.), i svaki model nudi različite prednosti, ali i slabosti.



Slika 1.1. Relaciona i nerelacione baze podataka

Unutar sistema, podaci su međusobno povezani, pa samim tim i razumevanje sistema zahteva obradu velikog broja konekcija, odnosno veza. Često su veze između podataka jednako bitne kao i sami podaci.

Graf baza podataka je jedan od modela NoSQL baza podataka, gde se podaci čuvaju pomoću graf strukture kao čvorovi i veze između čvorova umesto u vidu tabela ili dokumenata (Slika 1.2.).



Slika 1.2. Čvorovi i veze u graf bazi podataka

U relacionim bazama podataka se takođe mogu čuvati veze ili odnosi između entiteta, ali se dobijanje informacija o određenoj vezi vrši pretragom ili veoma skupom operacijom *JOIN*. Zbog toga se ispostavilo da relacione baze loše rukuju relacijama. Sa druge strane, u graf bazama podataka, se ne koriste spajanja ili pretrage da bi se dobile relacije. Relacije se čuvaju zajedno sa podacima na mnogo fleksibilniji način. Sve vezano za sistem je optimizovano za brzo kretanje kroz podatke.

Elementi modela graf baze podataka

Neo4j baza podataka se sastoji od sledećih elemenata:

- Čvorovi - reprezentuju entitete u grafu. Mogu se označiti labelom, koja predstavlja ulogu tog entiteta u sistemu. (npr. *Person, Movie..-Slika*). Čvorovi mogu da sadrže veći broj osobina i parova ključ-vrednost (npr *born:1967*-Slika).
- Relacije (veze) - obezbeđuju usmerene ili imenovane veze između dva ili više čvorova (npr *Person DIRECTOR Movie*). Relacije uvek imaju početni čvor, krajnji čvor, pravac i tip, ali mogu imati i svojstva baš kao i čvorovi. Iako su relacije usmerene, njima se može kretati i podaci se mogu dobiti iz bilo kog pravca. Jedan čvor može da ima neograničen broj veza ka drugim čvorovima, bez smanjenja performansi.
- Labele - Koriste se za grupisanje čvorova, a svakom čvoru se može dodeliti više labela. Oznake su indeksirane u cilju bržeg pronalaženja traženog čvora u grafu.
- Svojstva - To su atributi čvorova ali i relacija. Kako neo4j omogućava skladištenje podataka u vidu parova ključ vrednost, svojstva mogu imati bilo koju vrednost (*string, number ili boolean*).

Stavljanjem akcenta na vezu između podataka pre nego na podatke, grafovi su sjajni za velike, neuređene i povezane skupove opdataka. Sa njima je moguće postaviti kompleksne i apstraktne upite koji gledaju dalje od odnovne veze između podataka. Pored toga, moguće je vizuelno približiti sliku celog sistema.

Neo4j i prednosti Neo4j baze podataka

Neo Technology je kreirano Neo4j graf bazu podataka koja dovodi odnose između podataka u prvi plan. Neo4j je sistem otvorenog koda, NoSQL, graf baza koja je javno dostupna od 2007. Zasniva se na mrežno-orjentisanom modelu podataka sa osobinama u kojem su veze najvažniji objekti. Neo4j je izvorna graf baza podataka, što znači da implentira pravi model grafa sve do nivoa skladištenja, odnosno podaci se ne čuvaju kao "apstrakcija grafa" na drugoj tehnologiji, već se odmah čuvaju dok ih upisujemo. Ovo je razlog zbog koda Neo4j baza nadmašuje druge graf baze podataka i razlog zbog koga ostaje fleksibilna. Osim osnovnog grafa, Neo4j pruža sve ono što je očekivano od baze podataka: ACID(*atomicity, consistency, isolation, durability*) transakcije, podršku klastera i *runtime failover* (sigurna je, iako komponente od kojih je zavisna postanu nedostupne tokom izvršavanja). Fleksibilna graf šema, pruža laku izmenu i prilagođavanje tokom vremena, što omogućava a se kasnije dodaju novi odnosi kada se poslovne potrebe promene.

Neo4j omogućava :

- **Fleksiblina šema** – Nema potrebe za fiksnim modelom podataka. Atributi se mogu dodavati i uklanjati po potrebi.
- **Skalabilnost** – Omogućava povećanje broja upisa/čitanja kao i veličine same baze bez uticaja na brzinu izvršavanja upita.
- **Replikacija** – Omogućava replikaciju podataka obezbeđujući time maksimalnu sigurnost i pouzdanost.
- **ACID (Atomicity, Consistency, Isolation, Durability) model transakcija** - Neo4j u potpunosti podržava sva ACID pravila.
    - Atomiziranost (Atomicity) -Moguće je pokriti više operacija baze podataka u jednoj transakciji i provjeriti jesu li sve izvršene ispravno; ako jedna od operacija ne uspije, cijela će se transakcija neutralizirati
    - Dosljednost (Consistency) - Kada se upisuju podaci u Neo4j bazu podataka, garantirana je sigurnost da svaki klijent koji pristupi bazi podataka nakon toga čita najnovije ažurirane podatke.
    - Izolacija (Isolation) - Sigurno je da će operacije unutar jedne transakcije biti izolirane jedna od druge, tako da pisanje u jednoj transakciji neće utjecati na čitanje u drugoj transakciji.
    - Trajnost (Durability) -Podaci upisani u Neo4j bit će zapisani na disk i dostupni nakon ponovnog pokretanja baze podataka ili pada baze.
- **Ugrađena web aplikacija** – Upite za kreiranje i pretragu baze možete kreirati kroz jednostavan web interfejs, jer Neo4j obezbeđuje ugrađen *Neo4j Browser web* aplikaciju.
- **Jednostavno modeliranje (*Whiteboard Friendly*)** – Za Neo4j se kaže da je *whiteboard friendly* jer omogućava da model baze napravite jednostavnim crtanjem čvorova i veza.
- **Indeksiranje** - Podržano je indeksiranje korištenjem *Apache Lucence*.
- **Drajveri** - Postoje dodatni drajveri za rad sa Neo4j bazom podataka za mnoge programske jezike (Java, Spring, Python, Ruby, .NET, JavaScript…).

Neo4j je baza prilagođena čtanju. Ova baza je toliko brza, da omogućava slučajeve korišćenja grafova u realnom vremenu. Sa druge strane, pisanje u ovu bazu ima svoju cenu. Neki od razloga tome su:

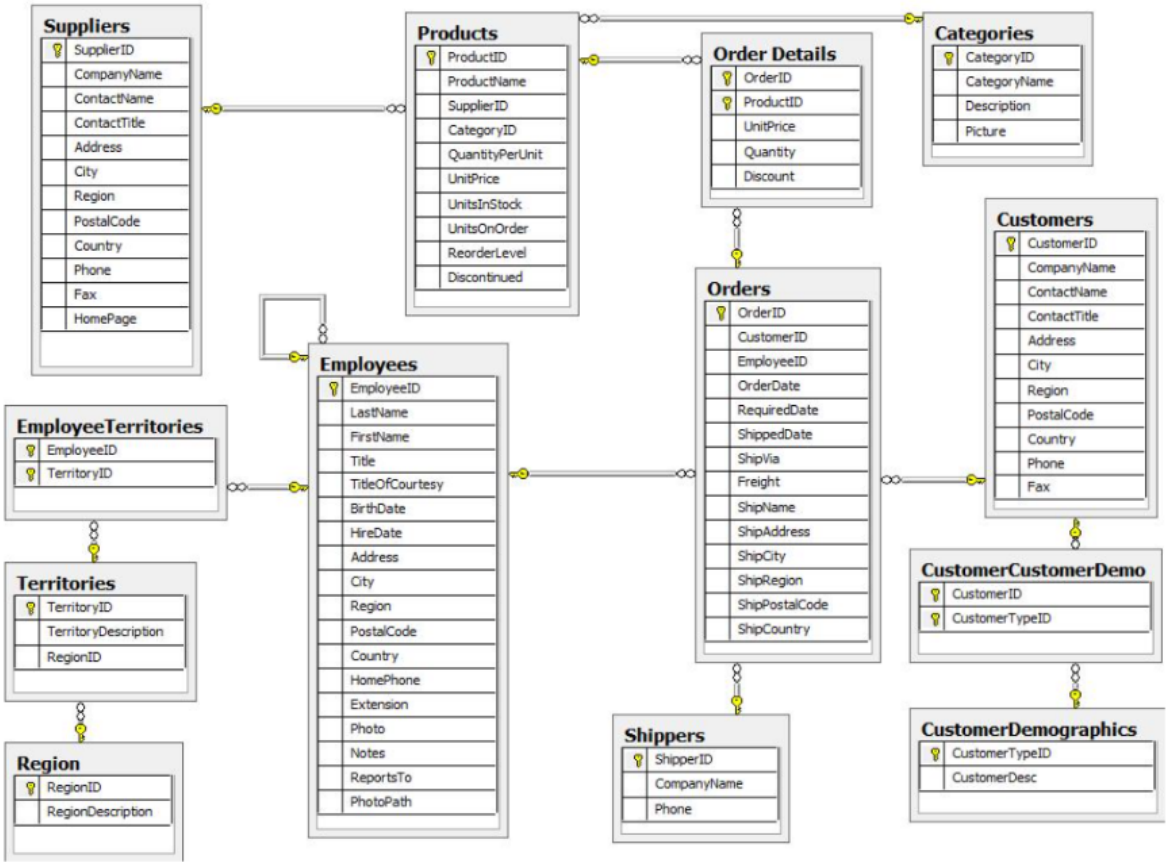
- Neo4j koristi *master* - *slave* arhitekturu i pisanje se uvek vrši na master, čak i u slučaju da se pokuša pisanje na *slave*-u, biće prebačeno na master.
- Svi podaci se nalaze na svakoj mašini, da bi se zaštitio referentni integritet. Kada upiti za pretragu skupa podataka postanu veći od RAM-a, sistem će postati značajno sporiji.
- Nije oprimizovana za pretragu, posebno u poređenju sa tehnologijama kao što su *elasticearch*.
- *Garbage collection* se zaustavlja. Moguće je alocirati previše memorije na *heap*-u i na taj način da ostanete bez memorije usled priliva velikog broja upita.

Prednosti i poređenja Neo4j baze podataka sa relacionim i nerelacionim bazama podataka

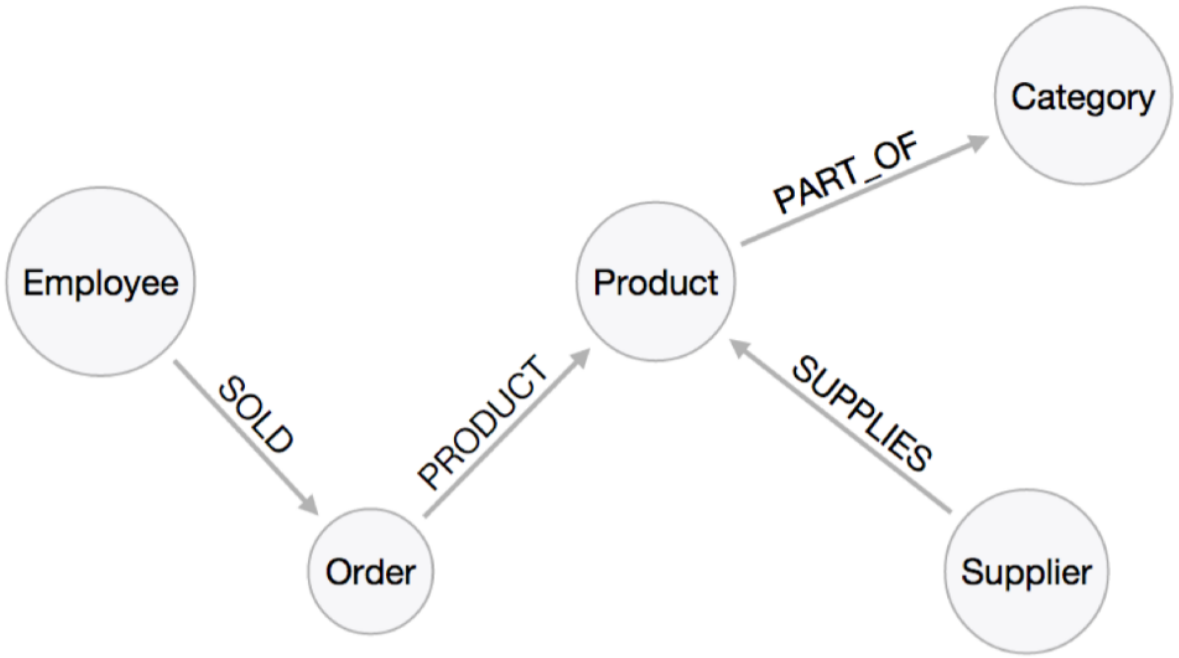
Posebno dizajnirana za rad sa ogromnim količinama povezanih podataka, Neo4j pruža sledeće prednosti:

- **Performanse** - U relacionim bazama podataka performanse trpe kako se broj i dubina odnosa povećavaju. U graf bazama podataka, kao što je Neo4j, performanse ostaju dobre iako količina podataka značajno poraste.
- **Fleksibilnost** - Neo4j je veoma fleksibilna, jer se struktura i šema grafa lako mogu prilagoditi promenama u aplikaciji. Takođe, moguće je lako nadograditi strukturu podataka, bez oštećenja postojeće funkcionalnosti.
- **Agilnost** - Struktura Neo4j baze podataka je laka za nadogradnju, tako da sama baza podataka može da se razvija zajedno sa aplikacijom.
- **Nema spajanja** - Neo4j ne zahteva složena spajanja za preuzimanje povezanih ili srodnih podataka, jer je lako dobiti detalje o njegovom narednom čvoru bez spajanja i indeksa.
- **Cypher upitni jezik** - Neo4j pruža deklarativni jezik upita za vizuelno predstavljanje grafa, kroisteći ascii-art sintaksu.
- **Visoka dostupnost** - Neo4j je visoko dostupan za velike poslovne aplikacije.

Složenost modela sistema koji imaju graf arhitekturu može biti značajno jednostavnija ukoliko je prikazana preko graf baze podataka(Slika 3.1. i Slika 3.2.).



Slika 3.1. Primer modela podataka u MySQL bazi podataka



Slika 3.2. Primer modela podataka u Neo4j bazi podataka

Cypher

Neo4j koristi deklarativan jezik koji služi za upite, *Cypher*, koji je sličan SQL-u, ali je optimizovan sa rad sa grafovima. Omogućava nam da navedemo šta želimo da selektujemo, dodamo, izmenimo ili obrišemo iz grafa bez eksplicitnog navođenja načina na koji će to biti učinjeno.

Kreiranje čvorova

```
CREATE (you:Person {name:"Jovica"}) RETURN you
```

Sledeći upit kreira novi čvor označen kao you tipa Person i sadrži podatak sa ključem name čija je vrednost „Jovica“.

Kreiranje relacija

```
MATCH(you:Person{name:"Jovica"})CREATE
(you)-[like:LIKE]->(neo:Database {name:"Neo4j" }) RETURN you,like,neo
```

Naredbom MATCH pronalazimo podatak tipa Person kome se vrednost ključa poklapa sa zadatom vrednosti „Jovica“ i dodeljuje mu oznaku you preko koje će se referincirati na taj podatak. Naredbom CREATE kreira se relacija LIKE ka drugom podatku tipa Database sa podatkom koji za ključ name ima vrednost „Neo4j“.

Brisanje čvora

```
MATCH (n:Useless) DELETE n
```

Brisanje relacija

```
MATCH (n { name: 'Andrew' }) - [r:FRIEND] -> () DELETE r
```

U običninm zagradama se navode vrsta čvora, npr. (p : Person), gde je “p” promenljiva tipa Person. Uglaste zagrade služe za navođenje veze između čvorova, npr [relatedTo : DIRECTED], gde relatedTo predstavlja promenljivu koja se odnosi na vezu “DIRECTED”. Takođe je moguće i specificirati koji paretri unutar određenog tipa, npr {title: “Cloud Atlas”}, gde ćemo dobiti sve filmove koji imaju naziv “Cloud Atlas”. Na Slici 4.1. su prikazani upit koji će vratiti sve osobe koje imaju neku vezu sa filmom koji se zove “Cloud Atlas”, pošto veza nije specificirana i vizuelni prikaz pretrage.

```
MATCH (p:Person)-[relatedTo]-(m:Movie {title: "Cloud Atlas"})
RETURN p, m, relatedTo
```

Slika 4.1. Rezultati pretrage u Neo4j bazi podataka

Cypher je lak za učenje, moćniji i koncizniji od SQL-a. Na primer, ako želimo da pronađemo sve glumce sa kojima radi Tom Hanks, *cypher* upit će izgledati :

MATCH (tom:Person {name: 'Tom Hanks'})-[a:ACTED_IN]->(m:Movie)<-[rel:ACTED_IN]-(p:Person)

return p, a, rel, m, tom

Za dobijanje istog rezultata, SQL upit će izgledati:

SELECT * FROM person p JOIN roles r ON(p.id = r.person_id)

JOIN movies m ON (m.id + r.movie_id)

JOIN roles r2 ON (m.id + r2.movie_id)

JOIN person p2 ON (p2.id = r2.person_id)

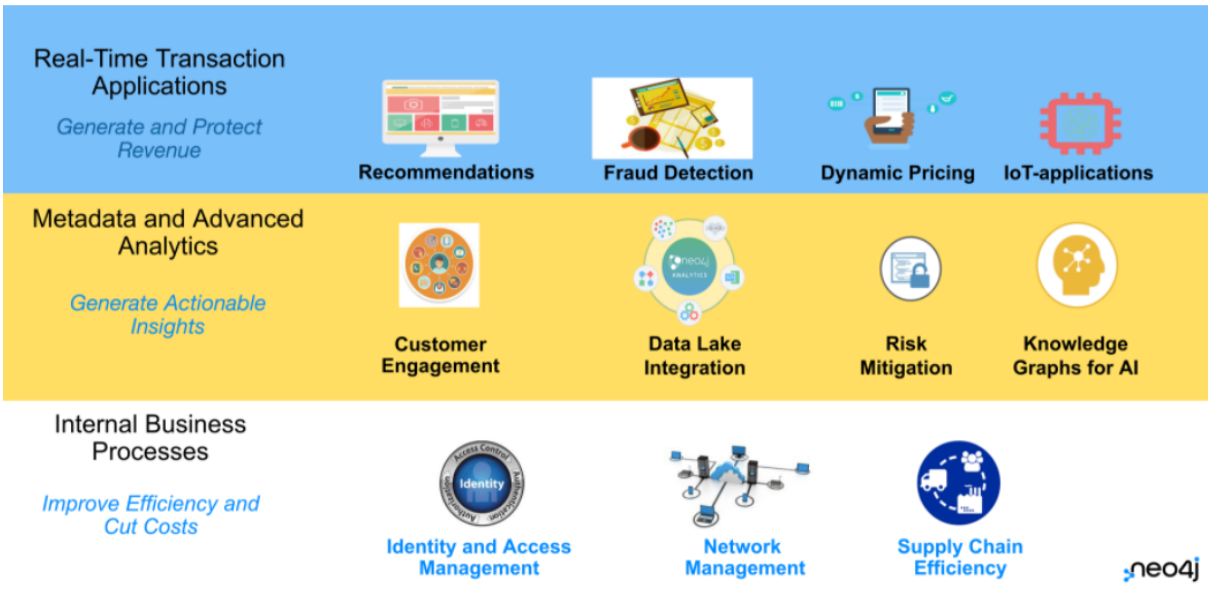
WHERE p2.name = “Tom Hanks”.

Primena

Neo4j koristi na hiljade startapova, obrazovnih institucija i velikih preduzeća u svim sektorima, uključujući finansije, vladu, energetiku, tehnologiju, maloprodaju i proizvodnju(Slika 5.1.).

Neki od slučajeva kada je najpogodnije koristiti Neo4j bazu podataka:

- **Otkrivanje i prevencija prevara** – Velike kompanije gube milijarde dolara godišnje zbog prevaranata koji se služe raznim sofisticiranim trikovima i prevarama kao što su, krađa identiteta, lažno predstavljanje, prevare sa kreditnim karticama i pranje novca. Neo4j pomaže u njihovom otkrivanju.
- **Praćenje mrežne infrastrukture**
- **Sistem za preporuke u realnom vremenu** – Posedujete *online* prodavnicu. Uz pomoć Neo4j baze podataka lako možete korisnicima preporučiti dodatne artikle iz ponude na osnovu onoga što pretražuju.
- **Društvene mreže** – Omogućava ubrzavanje kako razvoja tako i same aplikacije.
- **Prava pristupa i kontrola identiteta**



Slika 5.1. Slučajevi korištenja Neo4j baze podataka

Neo4j daje programerima i naucnicima pouzdane i precizne alate za brzu izgradnju inteligentnih aplikacija i radnih tokova mašinskog učenja.

Primer upotrebe Neo4j baze podataka za otkrivanje prevaranata

Jedan od načina na koji se problem sa prevarantima manifestuje prati sledeći scenario. Grupa prevaranata kreira u nekoj banci veliki broj bankovnih računa. Sve račune otvaraju koristeći kombinacije imena, prezimena, brojeva telefona, adresa,

brojeva socijalnog osiguranja itd. Nakon otvaranja, račune koriste normalno kako niko ne bi posumnjao na prevaru, što podrazumeva redovne prilive i odlive novca sa računa, javljanje na pozive bankarskih službenika, dostavljanje potrebne dokumentacije, primanje pošte koju banka šalje, itd. Nakon nekog vremena, svi računi odlaze u dozvoljeni minus i nestaju. Više se ne javljaju i banka ne može da stupi u kontakt sa njima. Dug se otpisuje i banka gubi ogromnu količinu novca.Neo4j pomaže u suzbijanju ovakvog problema pomoću sledećih klasa(Slika 5.2.):

```
// Create account holders
CREATE (accountHolder1:AccountHolder {
 FirstName: "John",
 LastName: "Doe",
 UniqueId: "JohnDoe" })

CREATE (accountHolder2:AccountHolder {
 FirstName: "Jane",
 LastName: "Appleseed",
 UniqueId: "JaneAppleseed" })

CREATE (accountHolder3:AccountHolder {
 FirstName: "Matt",
 LastName: "Smith",
 UniqueId: "MattSmith" })

// Create Address
CREATE (address1:Address {
 Street: "123 NW 1st Street",
 City: "San Francisco",
 State: "California",
 ZipCode: "94101" })
```

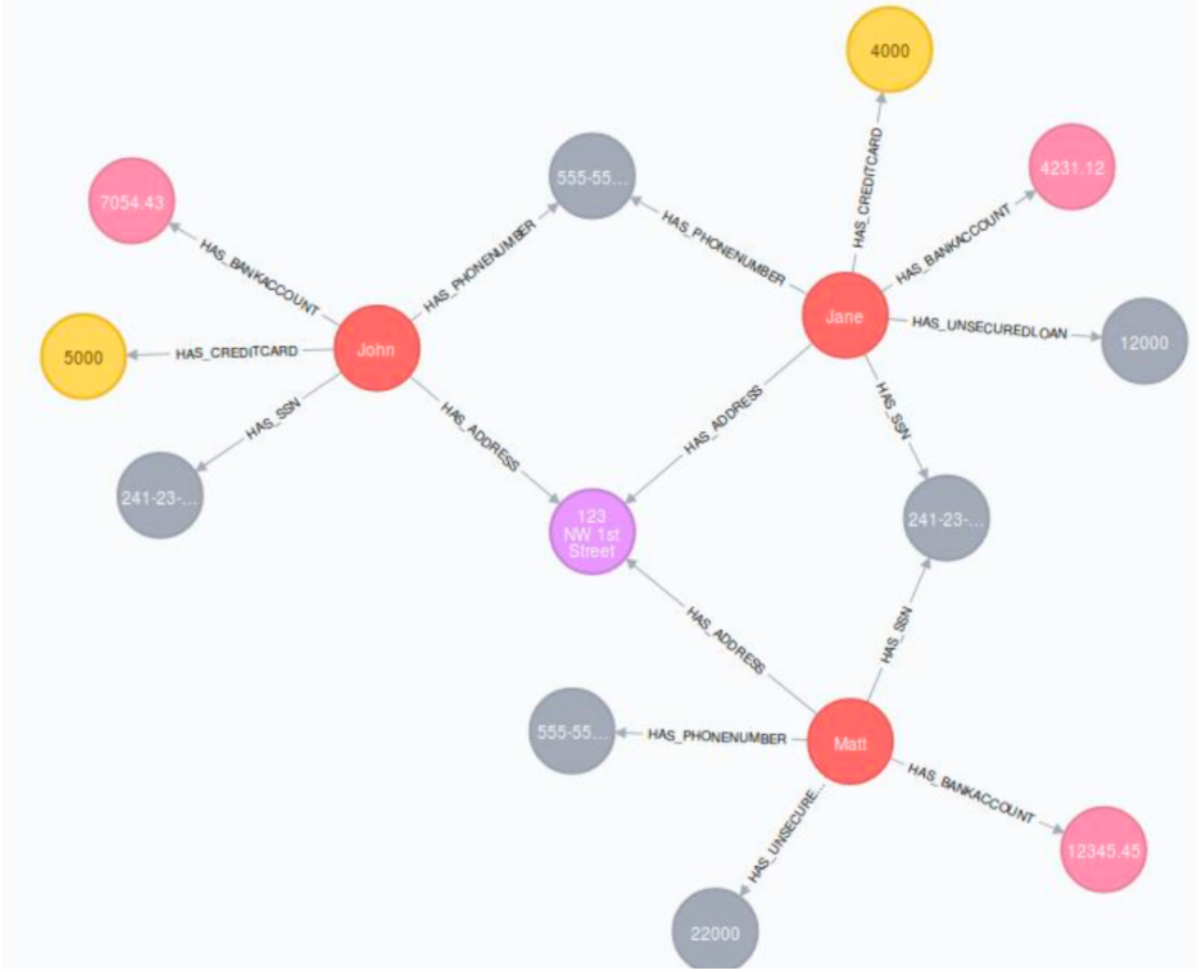
```
// Connect 3 account holders to 1 address
CREATE (accountHolder1)-[:HAS_ADDRESS]->(address1),
 (accountHolder2)-[:HAS_ADDRESS]->(address1),
 (accountHolder3)-[:HAS_ADDRESS]->(address1)

// Create Phone Number
CREATE (phoneNumber1:PhoneNumber { PhoneNumber: "555-555-5555" })

// Connect 2 account holders to 1 phone number
CREATE (accountHolder1)-[:HAS_PHONENUMBER]->(phoneNumber1),
 (accountHolder2)-[:HAS_PHONENUMBER]->(phoneNumber1)

// Create SSN
CREATE (ssn1:SSN { SSN: "241-23-1234" })

// Connect 2 account holders to 1 SSN
CREATE (accountHolder2)-[:HAS_SSN]->(ssn1),
 (accountHolder3)-[:HAS_SSN]->(ssn1)

// Create SSN and connect 1 account holder
CREATE (ssn2:SSN { SSN: "241-23-4567" })<-[:HAS_SSN]-(accountHolder1)
```

```
// Create Credit Card and connect 1 account holder
CREATE (creditCard1:CreditCard {
 AccountNumber: "1234567890123456",
 Limit: 5000, Balance: 1442.23,
 ExpirationDate: "01-20",
 SecurityCode: "123" })<-[:HAS_CREDITCARD]-(accountHolder1)

// Create Bank Account and connect 1 account holder
CREATE (bankAccount1:BankAccount {
 AccountNumber: "2345678901234567",
 Balance: 7054.43 })<-[:HAS_BANKACCOUNT]-(accountHolder1)

// Create Credit Card and connect 1 account holder
CREATE (creditCard2:CreditCard {
 AccountNumber: "1234567890123456",
 Limit: 4000, Balance: 2345.56,
 ExpirationDate: "02-20",
 SecurityCode: "456" })<-[:HAS_CREDITCARD]-(accountHolder2)

// Create Bank Account and connect 1 account holder
CREATE (bankAccount2:BankAccount {
 AccountNumber: "3456789012345678",
 Balance: 4231.12 })<-[:HAS_BANKACCOUNT]-(accountHolder2)
```

```
// Create Unsecured Loan and connect 1 account holder
CREATE (unsecuredLoan2:UnsecuredLoan {
 AccountNumber: "4567890123456789-0",
 Balance: 9045.53,
 APR: .0541,
 LoanAmount: 12000.00 })<-[:HAS_UNSECUREDLOAN]-(accountHolder2)

// Create Bank Account and connect 1 account holder
CREATE (bankAccount3:BankAccount {
 AccountNumber: "4567890123456789",
 Balance: 12345.45 })<-[:HAS_BANKACCOUNT]-(accountHolder3)

// Create Unsecured Loan and connect 1 account holder
CREATE (unsecuredLoan3:UnsecuredLoan {
 AccountNumber: "5678901234567890-0",
 Balance: 16341.95, APR: .0341,
 LoanAmount: 22000.00 })<-[:HAS_UNSECUREDLOAN]-(accountHolder3)

// Create Phone Number and connect 1 account holder
CREATE (phoneNumber2:PhoneNumber {
 PhoneNumber: "555-555-1234" })<-[:HAS_PHONENUMBER]-(accountHolder3)

RETURN *
```

Slika 5.2. Klase za otkrivanje prevaranata u Cypher upitnom jeziku

Rezultat izvršavanja ovog upita prikazan je na Slici 5.3.



Slika 5.3. Rezultati upita u Neo4j bazi podataka

Naredni korak predstavlja pronalazak vlasnika računa kojima se podaci poklapaju, što se obavlja narednim upitom.

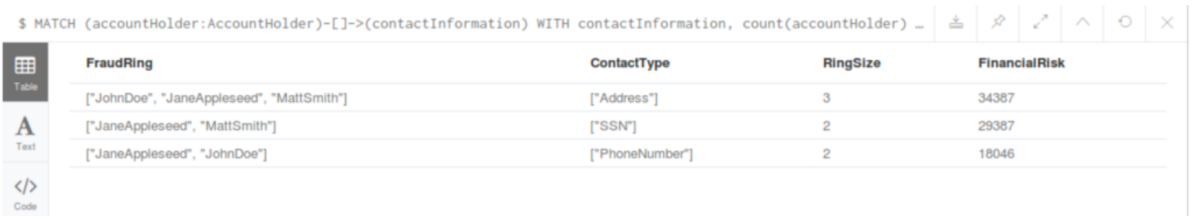
```
MATCH (accountHolder:AccountHolder)- []->(contactInformation) WITH
contactInformation, count(accountHolder) AS RingSize MATCH
(contactInformation) 1 RETURN AccountHolders AS FraudRing,
labels(contactInformation) AS ContactType, RingSize ORDER BY RingSize
DESC
```

Pored pronalaženja vlasnika računa koji su potencijalni prevaranti, treba otkriti i koliki maksimalan gubitak svaki od njih donosi ukoliko načini prevaru, što se obavlja narednim upitom:

```
MATCH (accountHolder:AccountHolder)-[]->(contactInformation) WITH
contactInformation, count(accountHolder) AS RingSize MATCH
(contactInformation)(unsecuredAccount) WITH collect(DISTINCT
accountHolder.UniqueId) AS AccountHolders,  contactInformation, RingSize,
```

SUM(CASE type(r)WHEN 'HAS_CREDITCARD' THEN unsecuredAccount.LIMIT WHEN 'HAS_UNSECUREDLOAN' THEN unsecuredAccount.Balance ELSE 0 END) AS FinancialRisk WHERE RingSize > 1 RETURN AccountHolders AS FraudRing,labels(contactInformation)ASContactType, RingSize,round(FinancialRisk) AS FinancialRisk ORDER BY FinancialRisk DESC

Krajnji rezultat primera može se prikazati u obliku sledeće tabele (Slika 5.4.)



Slika 5.4. Rezultati pretrage prikazani tabelarno u Neo4j bazi podataka

Performanse

Neo4j baza podataka konstantno teži poboljšanju performansi, pa su u verziji 3.5 performanse poboljšane na sledeće načine:

● Indeksiranje i pretraživanje cijelog teksta omogućuje brzo kretanje grafovskom bazom podatka pomoću pretraživanja teksta u oznakama, vrstama veza i vrijednostima svojstava

● Prošireno prirodno indeksiranje ubrzava unos podataka do 5x za sve vrste podataka (uključujući prostorne, vremenske i logičke vrijednosti)

● Smanjena su opterećenja kod pisanja zahvaljujući novom načinu za upravljanje transakcijama

● Poboljšanja preformansi prilikom sortiranja rezultata s indeksom ORDER BY upita u Cypheru

Neo4j distribuirana klaster arhitektura visokih performansi prilagođava se podacima i poslu, minimizirajući cenu i hardver, a maksimizira performanse u povezanim skupovima podataka. Sa Neo4j je moguće postići dobre performanse na milionima čvorova i trilionima konekcija između čvorova. Neo4j model podataka pruža izuzetno vredan psitup kada se gradi i razvija model podataka za promenljive poslovne zahteve. Pruža pouzdano brze transakcije sa visoko paralelizovanim protokom čak i kada podaci rastu, jer graf obezbeđuje susednost bez indexa, što skraćuje vreme čitanja i postaje još bolje kako složenost podataka raste.

U teoriji, graf baze podataka bi trebalo da budu značajno brže od relacionih baza podataka u obliku grafa (npr. pronalaženje svih prijatelja nekog prijatelja korisnika na društvenoj mreži, gde se pronalaženje usložnjava, odnosno traže se prijatelji od prijatelja od prijatelja…). Urađeno je istraživanje i vršeno poređenje između MySQL i Neo4j baze podataka, prilikom obrade 1 000 korisnika , gde svaki od njih ima oko 50 relacija (ukupno 50 000 relacija). Dobijeni rezultati za MySQL prikazani su na levoj slici (Slika 6.1.), dok su rezultati Neo4-aj prikazani na desnoj slici (Slika 6.2.).

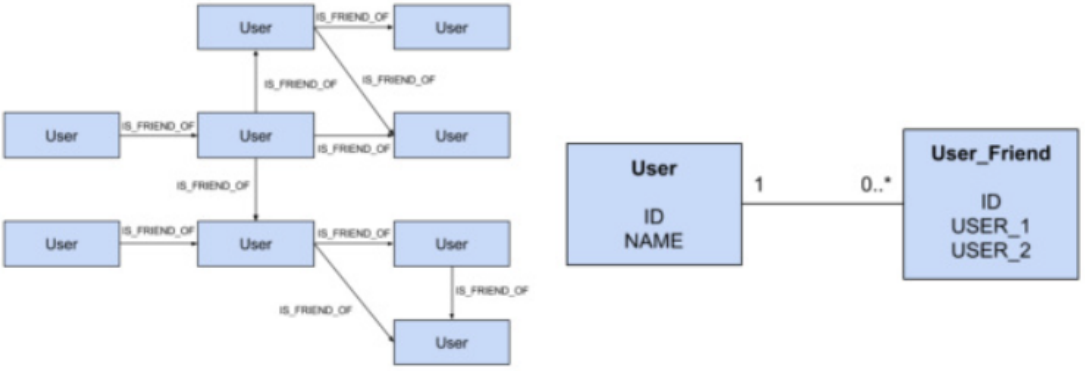
2	0.028
3	0.213
4	10.273
5	92.613

Slika 6.1. MySQL brzina pretrage

2	0.04
3	0.06
4	0.07
5	0.07

Slika 6.2 Neo4j brzina pretrage

Poređenjem ovih rezultata možemo uočiti da prilikom povećanja dubine odnosa, korištenjem Neo4j baze vreme obrade podataka se ne razlikuje značajno, ali istovremeno ova baza podataka postaje značajno brža od MySQL baze. Što je i očekivano, pošto se i sam model podataka unutar MySQL baze bnačajno komplikuje produbljavanjem odnosa.(Slika 6.2.).



Slika 6.2. Poređenje modela u graf i relacionoj bazi podataka

Neo4j u svetu velikih podataka

Neo4j ima neverovatnu mogućnost da kombinuje transformacije podataka i analizu podataka i rešava analitčke probleme sa kojima se relacione baze bore dugo. Neo4j je neverovatno brza prilikm čitanja podataka, ali ukoliko radimo sa sistemom koji podrazumeva velike količine upisa, ova baza može biti usko grlo u sistemu. Odnos između čvorova u *big data* sistemima ima veliki značaj, jer umesto da pružaju samo razumevanje vrednosti podataka, razjašnjava i odnos između podataka. Na primer, ako posmatramo analize *Walmart*-a, ova baza poadataka bi pomogla da se uoči vezu između između prodavaca koji su kupili lampe i čokoladice.

Organizacija se ne oslanja samo na podatke kada je u pitanju donošenje odluka. Ako želite da povećate svoju prodaju u npr. kljižari, nisu neophodni samo podaci o knjigama koje se prodaju, već je neophodno videti i konekciju između knjige i njenog kupca, odnosno koje knjige kupac obično kupuje i šta je zajedničko kupcima određene knjige. Neo4j baza podataka, može biti veoma efikasna kod pronalaženja ovakvih i mnogih drugih veza. Neo4j omogućava integraciju i sa drugim bazama podataka. U *big data* svetu, ova baza se široko primenjuje, a posebno je pogodna baza za rad sa društvenim mrežama, klasifikacijom specifičnih podataka ili pronalaženjem zajedničkih interesovanja i praksi. Kao najrasprostranjenija graf baza, Neo4j pomaže globalnim brendovima uključujući NASA-u, Volvo, eBAY i druge, da otkriju kako su ljudi, procesi i sistemi međusobno povezani. Aplikacije, napravljene pomoću Neo4j baze, rešavaju izazove kao što su veštačka inteligencija, otkrivanje prevara, preporuke u realnom vremenu, grafovi znanja i slično.

Skladište podataka kao što je Hadoop nije najbolje rešenje za sve big data probleme. Za probleme koji imaju arhitekturu mreže ili grafa, Neo4j može biti bolje rešenje. Takođe, krajnji rezultati u Hadoop-u nisu dobro predstavljeni korisnicima, zbog čega se Neo4j se često koristi za rad sa umreženim podacima i prikaz različitih skupova podataka. Podaci se u tom slučaju pripremaju u Hadoop-u, a zatim importuju u Neo4j bazu podataka da bi se omogućili upiti i vizualizacija. Dakle, Neo4j je veoma korisno kombinovati sa drugim skladištima podataka i koristiti kod obrade podataka sistema kod kojih su podaci veoma povezani ili imaju mrežnu arhitekturu.

Nadovezujući se na primer iz prethodnog poglavlja, gde je vršeno poređenje brzine Neo4j i MySQL baze podataka, na sledećoj slici će biti prikazano vreme potrebno za obradu velike količine podataka, kada umesto 50 000 konekcija, Neo4j baza obrađuje 50 miliona veza, za različite dubine.

2	0.01
3	0.168
4	1.359
5	2.132

Slika 7.1. Brzina pretrage u Neo4j bazi nad velikim skupom podataka

Zaključak

Neo4j je jedna od vodećih svetskih graf baza podataka otvorenog koda, razvijena upotrebom Java i Scala tehnologije, koja osigurava ACID kompatibilan transakcioni backend za aplikacije. Kao najrasprostranjenija deplojovana graf baza na svetu, neo4j omogućava prikaz veze između ljudi, procesa i sistema. Neo4j želi pružiti mogućnosti rješavanja mnogih različitih vrsta poslovnih i tehničkih problema. Bilo da se koriste grafovi za transakcije, analizu tržišta, optimizaciju poslovanja i slično, Neo4j nastoji pružiti besprekoran proces integriranja alata s ostatkom postojećeg sistema. Neo4j koristi Cypher jezik za vršenje upita nad grafom.

Ova baza podataka je odlična za rad sa podacima koji imaju mrežnu strukturu, odnosno podaci među kojima su odnosi veoma važni. Ovu bazu je efikasno koristiti prilikom rada sa velikim skupovima podataka, češće kao pomoćnu bazu, koja će doprinositi boljoj vizualizaciji ili analizi podataka.

Neo4j pruža brojne alate i biblioteke kako bi razvoj bio lakši i brži. Ova baza podataka pruža i alate za programere kao što su Neo4j Desktop, Browser i Sandbox koje olakšavaju razvoj i prikaz grafički prikaz.

Literatura:

1. Why graphs are so effective in big data analytics? - https://www.cleverism.com/graph-databases-effective-big-data-analytics/
2. Neo Technology CEO: What is a graph database... and why big data needs one - https://www.computerweekly.com/blog/CW-Developer-Network/Neo-Technology-CEO-What-is-a-graph-database-and-why-big-data-needs-one
3. Neo4j - Relational Databases vs Graph Databases: A Comparison - http://neo4j.com/developer/graph-db-vs-rdbms/
4. How much faster is a graph database, really? - https://neo4j.com/news/how-much-faster-is-a-graph-database-really/
5. Neo4j what you need to know - https://www.tutorialspoint.com/neo4j/neo4j_overview.htm
6. Challenges of neo4j at the heart of software - https://dev.to/l04db4l4nc3r/challenges-of-neo4j-at-the-heart-of-software-2dbi
7. Capabilities of the Neo4j Graph Database- https://rubygarage.org/blog/neo4j-database-guide-with-use-cases
8. Grafovske baze podataka - https://blog.imi.pmf.kg.ac.rs/grafovske-baze-podataka-neo4j/