Jihočeská univerzita v Českých Budějovicích
Přírodovědecká fakulta

Grafové databáze a jejich aplikace na sociální sítě

Bakalářská práce

Tomáš Bartha

Školitel: doc. Ing. Ladislav Beránek, CSc., MBA

České Budějovice 2015

Prohlášení:

Prohlašuji, že svoji bakalářskou práci jsem vypracoval samostatně pouze s použitím pramenů literatury uvedených v seznamu citované literatury.

Prohlašuji, že v souladu s § 47b zákona č. 111/1998 Sb. v platném znění souhlasím se zveřejněním své bakalářské práce, a to v nezkrácené podobě elektronickou cestou ve veřejně přístupné části databáze STAG provozované Jihočeskou univerzitou v Českých Budějovicích na jejích internetových stránkách, a to se zachováním mého autorského práva k odevzdanému textu této kvalifikační práce. Souhlasím dále s tím, aby toutéž elektronickou cestou byly v souladu s uvedeným ustanovením zákona č. 111/1998 Sb. zveřejněny posudky školitele a oponentů práce i záznam o průběhu a výsledku obhajoby kvalifikační práce. Rovněž souhlasím s porovnáním textu mé kvalifikační práce s databází kvalifikačních prací Theses.cz provozovanou Národním registrem vysokoškolských kvalifikačních prací a systémem na odhalování plagiátů.

V Českých Budějovicích dne

Bartha T., 2015: Grafové databáze a jejich aplikace na sociální sítě [Graph databases and their applications to social networks. Bc. Thesis, in Czech.] – 37 p., Faculty of Science, University of South Bohemia, České Budějovice, Czech Republic

Poděkování

Děkuji panu doc. Ing. Ladislavu Beránkovi, CSc., MBA., za odborné vedení mé bakalářské práce a pomoc při jejím zpracování.

Anotace

Hlavní myšlenkou této bakalářské práce je analýza NoSQL databází se zaměřením na grafové databáze a jejich použití v případě sociálních sítí. S použitím technologie Neo4j je implementována vzorová databáze a je předvedeno využití dotazovacího jazyka Cypher. V poslední části bakalářské práce je provedena analýza dat vybrané vzorové databáze pomocí Neo4j Serveru a jazyka Cypher a provedeno zhodnocení této analýzy z hlediska využití.

Abstract

Main idea of the bachelor work is theoretically introduce NoSQL databases and mainly graph databases with their using at social network. Sample database is implemented with using Neo4j technology and show Cypher query language use cases. At the last part we will analyze sample database data with using Neo4j server and query language and perform analysis assessment.

Obsah

1 Úvod

Cílem práce je analyzovat funkce a možnosti použití grafových databází a provést vzorovou analýzu dat sociální sítě. Před samotnou analýzou se však zaměříme na pojmy NoSQL databáze, grafové databáze a sociální sítě.

Na počátku práce si nejdříve představíme skupinu databází nabízející alternativní způsoby ukládání dat na rozdíl od relačních databází, NoSQL databáze. Postupně se zaměříme na specifický typ NoSQL databází, grafové databáze a vysvětlíme si, jak se v těchto databázích modelují data.

Po vysvětlení pojmu sociální síť a vysvětlení jejich návaznosti na grafové databáze se přesuneme ke konkrétnímu typu grafových databází Neo4j. Neo4j je nejpopulárnější grafovou databází na trhu a má širokou základnu fanoušků z řad vývojářské komunity i široké veřejnosti. Podíváme se na hlavní přednosti Neo4j , jejich dotazovací jazyk Cypher a webové rozhraní Neo4j serveru.

V poslední části práce se zaměříme na data z bitcoinového serveru Blockchain, která svojí strukturou připomínají data sociálních sítí. Na těchto datech provedeme analýzu financí za účelem identifikování financí získaných nelegální činností prostřednictvím šíření počítačového viru. Po získání výsledků analýzy zhodnotíme námi získaná data z hlediska použití.

2 Cíle práce a použitá metodika

2.1 Cíle práce

Cílem práce je provést analýzu grafových databází za účelem výběru nejvhodnější z nich. Vybrat jednoho zástupce, pomocí kterého provedeme analýzu dat databáze s následným zhodnocením dosažených výsledků analýzy.

2.2 Metoda řešení

Práce je rozdělena na dvě důležité části, jednotlivými kroky v každé z těchto částí je směřováno k pochopení zákonitostí kolem tématu grafových databází a možnostem jejich praktického využití. V tematické části se práce zabývá důležitými pojmy z oblastí databází a sociálních sítí. V části praktické je kladen důraz na technické řešení tématu, instalaci serveru grafové databáze, implementaci dat do databáze a analýzu dat.

Předpokládaný přínos

Přínos práce vidím hlavně v osobním zvýšení odbornosti v tématu grafových databází a získání zkušeností s prací v konkrétním prostředí databáze. Práce může být vhodná pro čtenáře bez znalostí v problematice a může sloužit čtenáři k osvojení základních technik práce s databází, dotazovacím jazykem a provedení analýzy. Do budoucna by mohla sloužit jako základ pro širší analýzu dat nebo například jako vzor pro aplikaci umožňující automatizaci analýzy.

3 NoSQL Databáze

Relační databáze jsou dlouho používaným a velmi rozšířeným řešením v oblasti databázových systémů. S pokrokem v oboru technologií, narůstajícím početním výkonem a množstvím dat, které je potřeba zpracovat se začala hledat odpovídající alternativa k relačním databázím. Touto alternativou jsou NoSQL databáze.

NoSQL databáze nepoužívají primárně jazyk SQL jako relační databáze, avšak neznamená to, že by všechna datová úložiště odmítala jazyk SQL, jak by mohl název napovídat, proto pro databáze používáme vysvětlení „Not only SQL“. Hlavní přednosti NoSQL databází jsou jednoduchý datový model databáze, rychlá odezva a výkon a velká škálovatelnost. Právě výkon databáze je pro NoSQL důležitější než její konzistence. Protože se u NoSQL databází neřeší transakční zpracování (ACID), referenční integrita atd. je jejich rychlost při vykonávání dotazů lepší než u robustních relačních databází. Popišme si důležité pojmy z oblastí databází, které jsou ACID, CAP teorém a BASE.

3.1 ACID

ACID je složenina z anglických slov Atomicity, Consistency, Isolation a Durability. Jde o základní funkcionalitu databázového systému, která zajišťuje, aby data byla konzistentní a v případě chyby se dala vrátit do původního stavu.

Hlavními vlastnostmi tohoto konceptu jsou:

- Atomicity (Nedělitelnost) - Nedělitelnost, je důležitá vlastnost, která zajišťuje provedení transakce jako celku. Pokud není transakce provedena jako celek, není provedena vůbec a systém provede odezvu (chybové hlášení)
- Consistency (Konzistentnost) - Každá transakce kterou provedl databázový systém, musí být převedena z jednoho konzistentního stavu do druhého. Jen taková data, která splňují všechna integritní omezení, mohou být zapsána do databáze.
- Isolation (Izolovanost) - Všichni uživatelé vidí až do ukončení transakce data v původní podobě, jako byla před začátkem transakce. Pokud je vrácením transakce zasažena i jiná transakce, musí být vrácena i tato transakce.
- Durability (Trvanlivost) - Všechny změny uložené do databáze po provedení úspěšné transakce musí být trvalé a nemohou být ztraceny.

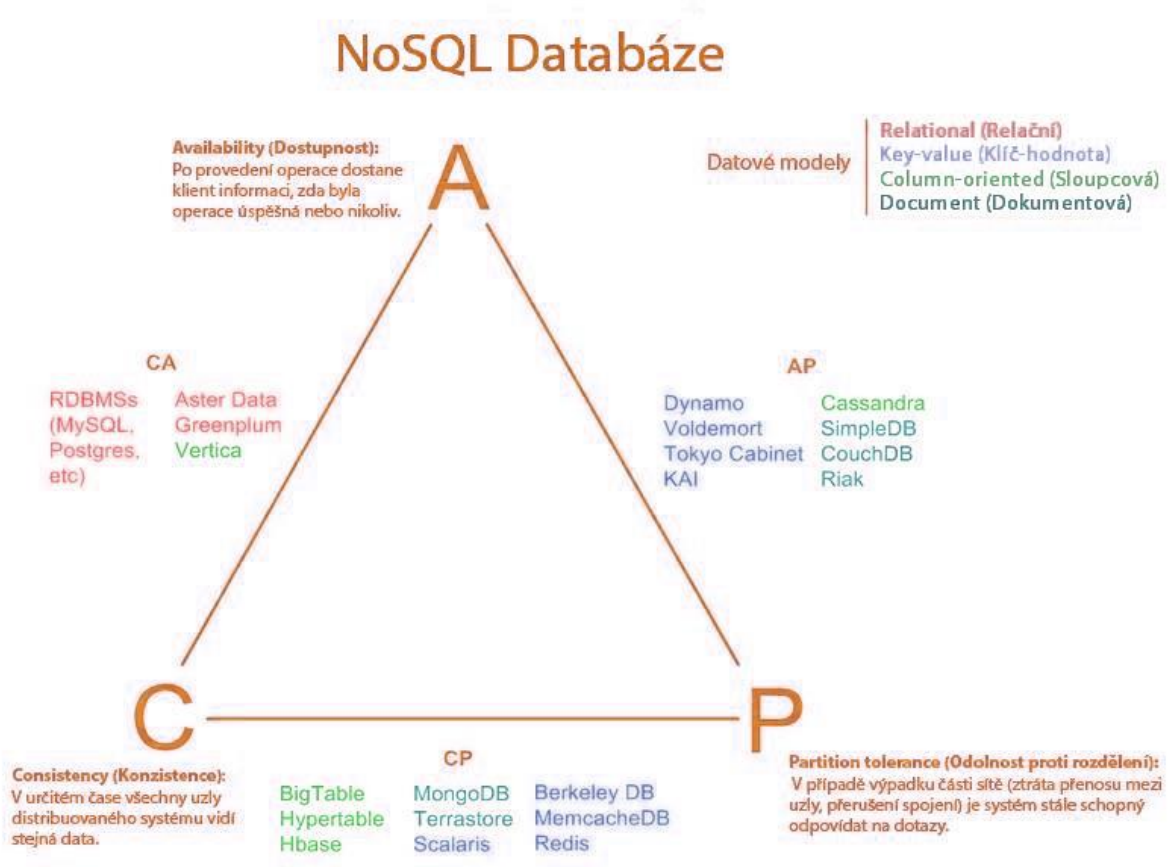
3.2 CAP teorém

Podle CAP teorému máme 3 základní požadavky na distribuovaný systém, konzistence, dostupnost, odolnost proti výpadku sítě. CAP teorém říká, že neexistuje takový distribuovaný systém, který by dosáhl na všechny tři požadavky najednou.

Vysvětleme si význam těchto požadavků:

- Consistency (Konzistence) - V určitém čase všechny uzly distribuovaného systému vidí stejná data.

- Availability (Dostupnost) - Po provedení operace dostane klient informaci, zda byla operace úspěšná nebo nikoliv.
- Partition tolerance (Odolnost proti rozdělení) – V případě výpadku části sítě (ztráta přenosu mezi uzly, přerušení spojení) je systém stále schopný odpovídat na dotazy.



Obrázek 1 - Přístup k CAP teorému[1]

Ačkoliv jsou u NoSQL databází žádoucí všechny tři vlastnosti, je možné dosáhnout pouze dvou z nich. Proto je třeba při nasazování databáze zhodnotit požadavky aplikace a rozhodnout se pro přístup AP nebo CP

3.3 BASE

BASE jako akronym je používána k popsání vlastností určitých databází, obvykle jsou to databáze NoSQL, a je často označován jako opak ACID. Pokud si zkratku BASE rozložíme, dostaneme se ke slovům **B**asically **A**vailable, **S**oft state, **E**ventual consistency.

- Basically Available – aplikace pracuje bez přerušení
- Soft state – nemusí být v každém okamžiku konzistentní
- Eventual consistency – Aplikace se musí vždy navrátit do konzistentního stavu

[1] Obr. 1 – Zdroj: http://blog.nahurst.com/visual-guide-to-nosql-systems

V případě ACID modelu má větší důležitost konzistence nad dostupností, pro BASE platí opačný stav, tedy dostupnost nad konzistencí.

3.4 Přehled typů NoSQL Databází

NoSQL databáze poskytuje mechanismy pro ukládání a načítání dat, která jsou modelována jiným způsobem než tabulkovým, používaným v relačních databázích. Motivacemi k tomuto přístupu byly hlavně jednoduchost konstrukce, horizontální škálování a kontrola nad dostupností. Datová struktura se liší v jednotlivých případech od RDBMS (např. stromy, grafy, klíč-hodnota) a to vede k tomu, že některé operace jsou rychlejší v NoSQL. [1]

V případě NoSQL databází rozeznáváme 4 hlavní typy databází:

3.4.1 Key-value database (Klíč-hodnota)

Úložiště typu Key-value je pravděpodobně nejjednodušší formou DBMS[2], jedná se o typ NoSQL databáze která je zproštěna schématu. Data do úložiště jsou ukládána v podobě klíče jako jeden sloupec a hodnoty jako druhý sloupec. Klíč může být umělý nebo generován automaticky zatímco hodnota může být např. typu string, JSON, BLOB. Databáze prakticky nemá ponětí o tom, co se uchovává v „hodnotě“, proto tato logika musí být řešena na aplikační vrstvě. Tento typ databáze obvykle k vyhledávání používá hash tabulku ve které existují unikátní klíče a ukazatele na konkrétní položky dat. Díky této metodě je dosahováno velké rychlosti při vyhledávání.

Příklady: Cassandra, Redis, Big Table, …

3.4.2 Column Oriented Database (Sloupcová)

Sloupcově orientované databáze byly vytvořeny pro ukládání a zpracování velmi velkých objemů dat distribuovaných mezi velké množství serverů. V případě sloupcově orientovaných databází jsou data uchovávána v buňkách (tabulkách) seskupených ve sloupcích dat místo obvyklého skladování v řádcích dat. Na rozdíl od RDBMS[3] může mít každý řádek jiný počet sloupců. Sloupce jsou logicky seskupené do takzvaných „Column Families“, které mohou obsahovat neomezený počet sloupců a obsahují zpravidla data stejného typu. Sloupec jako takový je pouze pár typu klíč-hodnota, který existuje v „Column Family“.

Příklady: MonetDB, Vertica, …

3.4.3 Document Oriented Database (Dokumentová)

Dokumentově orientovaná úložiště jsou svými principy velmi podobná úložištím typu Klíč-hodnota, avšak hlavním rozdílem mezi nimi je, že hodnota uložená v úložišti má určitou strukturu a kódování spravovaných dat. Jako běžné standardy kódování jsou používány např. JSON, XML a binární formáty (jako PDF a MS Word). U dokumentových databází je

[2] DBMS – database management system
[3] RDBMS – relations database management system

výhodou načtení metadat provázaných s obsahem dokumentu a díky tomu lze uložený obsah třídit do podobných skupin na základě obsahu, což značně usnadňuje vyhledávání.

Příklady: CouchDB, MongoDB, Terrastore, …

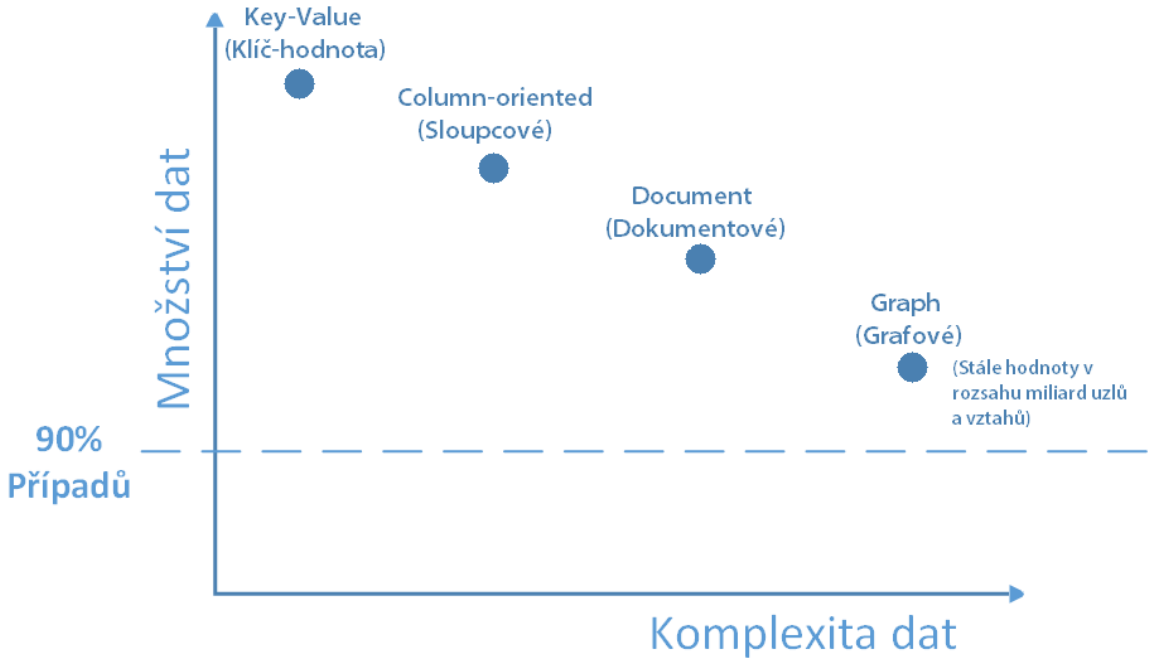
3.4.4 Graph database (Grafová)

Grafové databáze nepoužívají pevný formát jako u SQL, tabulkového a sloupcového znázornění, ale místo toho je použito flexibilní grafické znázornění. Grafické znázornění je tvořeno sítí uzlů (vrcholů) a hran. Každý uzel představuje objekt a každá hrana představuje vazbu mezi objekty. Uzly i hrany mohou mít své atributy. Grafovými databázemi se budeme zabývat v následujících částech práce.

Příklady: Neo4j, FlockDB, AllegroGraph, …

3.5 Porovnání typů NoSQL databází:

Pokud porovnáme typy NoSQL databází vzhledem k složitosti dat a schopnosti databáze tato data uložit, dojdeme k závěrům patrným z obrázku. Databáze typu Key-value jsou schopny pojmout největší množství dat, ale tyto data mají nejmenší složitost. Na opačné straně jsou Grafové databáze, jejichž složitost je velká, ale pojmou z NoSQL nejmenší množství dat. [2]



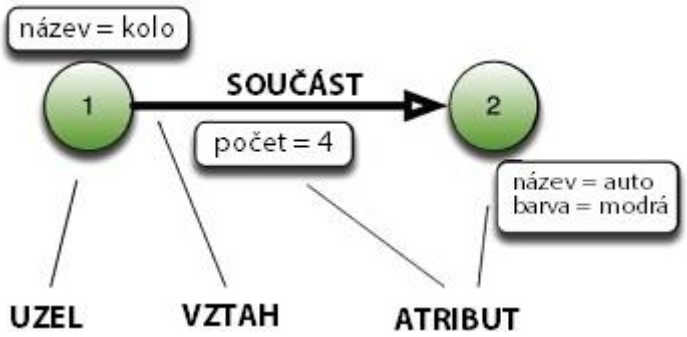
Obrázek 2 - Graf poměru komplexity a objemu NoSQL databází[4]

[4] Obr. 2 Zdroj: http://www.3pillarglobal.com/sites/default/files/nosql-3a.png

4 Grafové databáze

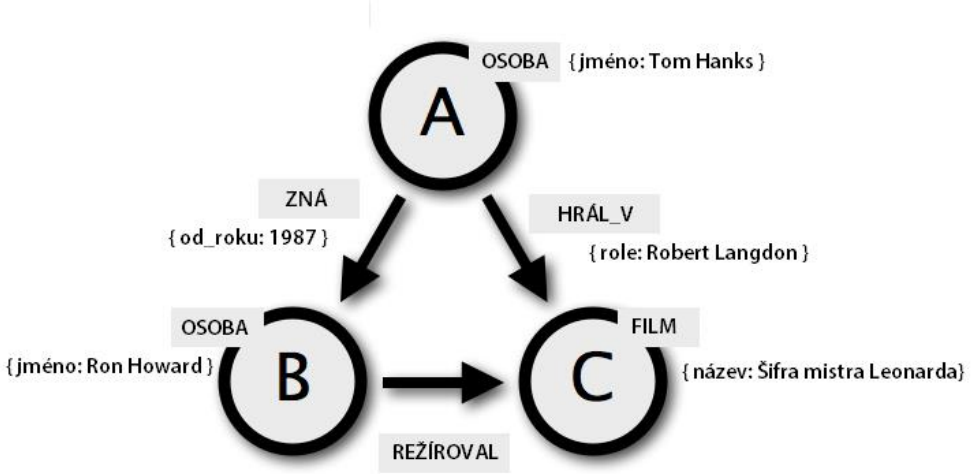
Podle obecné definice, jsou grafové databáze jakýkoliv systém, který poskytuje „bezindexovou“ sousednost, tj. každý vrchol obsahuje přímé odkazy na své sousedy a tím odpadá nutnost používání indexů. [3]

Grafové databáze nemají tak rigidní formát jako SQL nebo tabulkové či sloupcové modely, na rozdíl od nich je grafový model flexibilnější k řešení problému se škálovatelností. Grafové struktury využívají hrany, uzly a jejich vlastnosti k prezentaci uložených dat. Všechny uzly jsou organizovány vztahy mezi jinými uzly, k čemu se využívají hrany mezi těmito uzly.



Obrázek 3 - Struktura Grafové databáze[5]

Tento typ databází je velice vhodný pro ukládání projektů, u kterých nám velmi záleží na propojenosti dat a nahrazují v těchto případech relační databáze, které jsou pro tyto případy nepřirozené vzhledem k nemožnosti obejít se bez početně náročných operací JOIN. Grafové databáze nalézají hlavní použití hlavně v oblastech sociálních sítí, bioinformatice, telekomunikačních systémech, elektronickém obchodování apod.



Obrázek 4 - Příklad modelu Grafové databáze[6]

[5] Obr. 3 Zdroj: http://www.infoq.com/resource/articles/graph-nosql-neo4j/en/resources/image3.jpg
[6] Obr. 4 Zdroj: http://neo4j.com/graphacademy/online-course/

Na obrázku 4 lze vidět jednoduchý model grafové databáze obsahující několik uzlů a hran. Každý uzel obsahuje atributy a mezi dvěma vrcholy existuje alespoň jedna hrana, která může také být popsána atributy. Vlastnosti lze přirovnat k atributům relačních databází.

5 Data vhodná pro zpracování pomocí grafové databáze

Pomocí grafové databáze je vhodné zpracovávat strukturovaná data, u kterých existuje mezi daty vztah. Ten může vyjadřovat propojení např. mezi uzly počítačové sítě, odkaz na webové stránky a další. Velkou oblastí, kde jsou generována data vhodná pro zpracování grafovými databázemi, jsou sociální sítě.

5.1 Sociální síť:

Pojem sociální síť vychází původně ze sociologie, kde představuje skupinu lidí, která spolu udržuje komunikaci různými prostředky. [4]

V prostředí informatiky sociální síť představuje webovou platformu založenou na komunitě uživatelů a institucí vzájemně propojených vazbami. [5] Tyto vazby mohou mít různý význam, obvykle nám znázorňují společné zájmy či přátelství. Uživatelé, lidé nebo firmy, se nejčastěji prezentují světu svou identitu nejčastěji prostřednictvím profilu nebo stránky. Profily a stránky jsou spolu vzájemně propojeny a interagují a sdílí informace veřejnou, částečně veřejnou nebo privátní formou. Informace obvykle mají formu textu nebo multimediálních souborů jako fotky, obrázky, video a audio.

5.1.1 Analyzování sociální sítě

Sociální síť jako taková může být mapována a analyzována. Z hlediska informací generuje sociální síť velké množství dat (Big Data), která po jejich analyzování mohou poskytovat informace vhodné pro podnikání, vědecké účely apod. Obecně rozšířenými metodami jsou analýza sociální sítě a sociometrie.

Výsledkem analýzy sociální sítě je mapa sociální sítě, která graficky znázorňuje všechny prvky sociálního systému a vztahy mezi nimi. Speciálním případem analýzy sociální sítě je sociometrie, jejím smyslem je zmapovat všechny vzájemné vztahy poté je rozdělit na vztahy kladného, nulového a negativního směru. Po zpracování těchto vztahů do grafické podoby vzniká sociogram, který znázorňuje vzájemné vztahy členů zkoumané sociální sítě a jejich kvalitu. [6]

5.2 Druhy sociálních sítí

Facebook

Sociální síť založená roku 2004 Markem Zuckerbergem je momentálně se 1,4 miliardy aktivních účtů největší a nepopulárnější sociální sítí na světě. [7] Facebook vznikl původně pro účely studentů Harvardu, později se rozšířil mezi ostatní univerzity a vzhledem k jeho popularitě se roku 2006 stal dostupným i pro širokou veřejnost.

Vzhledem k velikosti a počtu uživatelů Facebooku je využíváno technologií upravených a přizpůsobených požadavkům na dostupnost a rychlost odezvy. Jednou z těchto technologií je i NoSQL databáze Cassandra používaná pro prohledávání poštovních zpráv Facebooku.

Google+

Jeden z přímých konkurentů Facebooku a Twitteru, sociální síť provozovaná společností Google od roku 2011. Momentálně se pohybuje okolo 300 milionů aktivních účtů. [7]

Google+ umožňuje sdílení obsahu mezi uživateli, skupinami uživatelů nebo skupinami se stejným zájmem, tím Google využívá množství vztahů mezi uživateli, které nemusí být jen vztahem „přátelí se“.

Twitter

Twitter je mikroblogovací sociální síť umožňující uživatelům odeslat krátké 140 znakové zprávy nazvané „tweety“. Twitter byl založen roku 2006 Jackem Dorseym, Noahem Glassem, Bizem Stonem a Evanem Williamsem. V dnešní době má Twitter přibližně 288 milionů aktivních účtů [7] a více než 500 milionů účtů celkem.

Twitter rozvíjí vlastní NoSQL databázi nazvanou Manhattan, která splňuje nároky na funkcionalitu. Pro uchovávání grafových dat je u Twitteru používána FlockDB grafová databáze.

Youtube

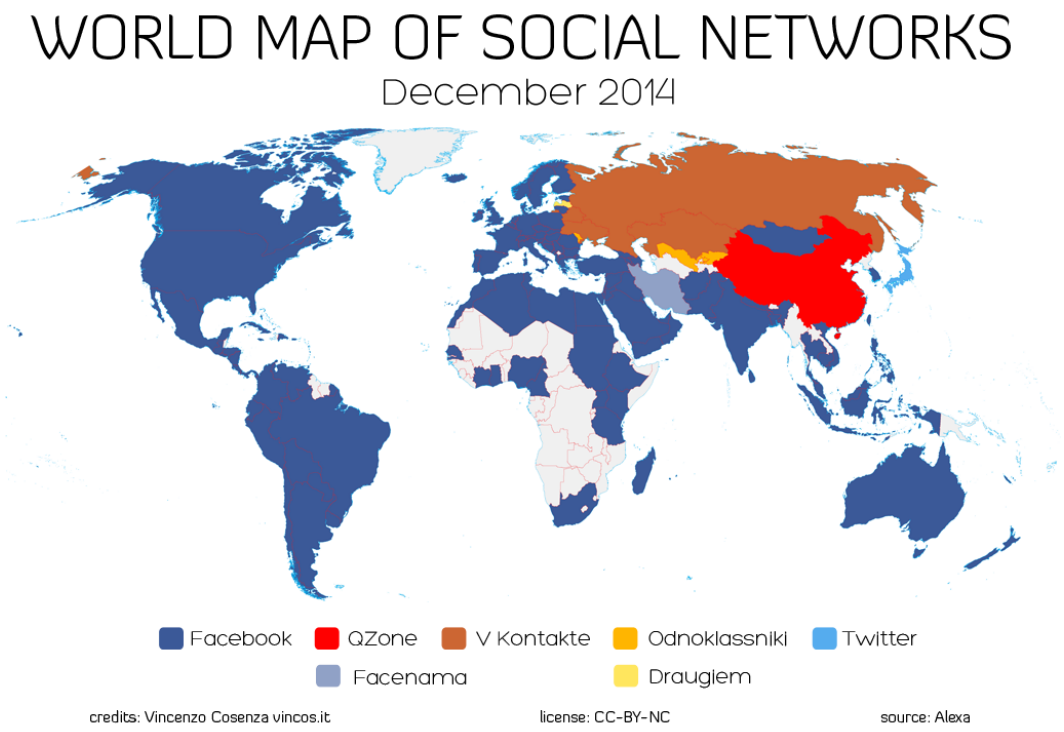
Youtube byl založen roku 2005 Chadem Hurleyem, Steve Chenem a Jawedem Karim jako internetový server pro sdílení videosouborů. Na konci roku 2006 byl zakoupen Googlem a v dnešní době je celosvětově rozšířen a je největším serverem pro sdílení videosouborů. Registrovaní uživatelé mohou nahrávat, komentovat a „likovat“ videa, proto se dá i youtube považovat za sociální síť.

Lidé.cz

Lidé.cz je internetová seznamka a chat spuštěná firmou Seznam.cz. Od roku 2008 byla služba prezentována více jako sociální síť, ale v roce 2014 byla změněna

funkčnost služby a jejích funkcionalit, následně byl celý portál prezentován více jako seznamka a chat. V dnešní době je stále jednou z nejnavštěvovanějších českých sociálních sítí.

Na obr. 5, lze vidět momentální rozložení sil sociálních sítí ve světě.

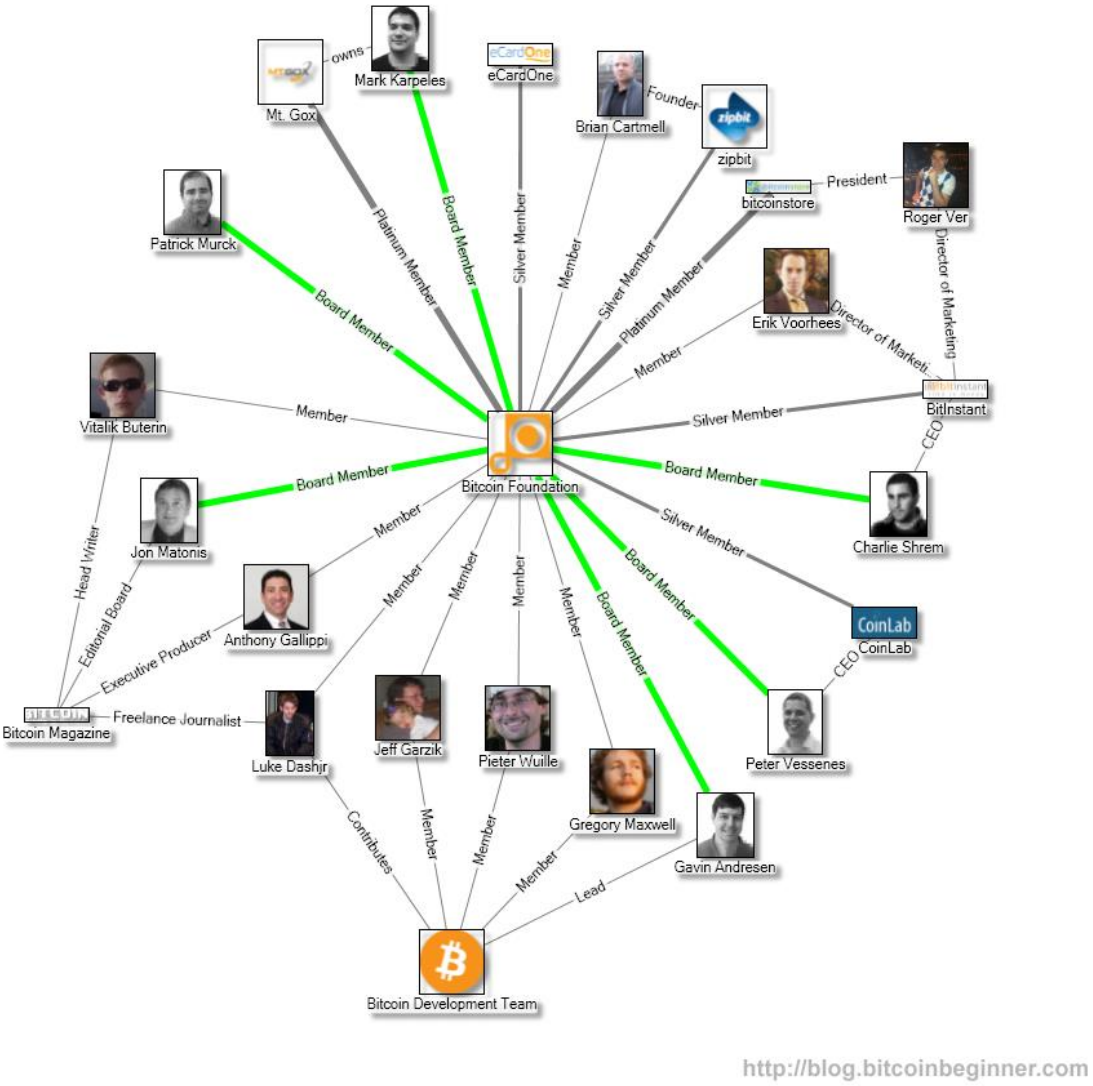


Obrázek 5 - Používání sociálních sítí ve světě[7]

5.2.1 Další sociální sítě

Existuje dále celá řada sociálních sítí nejrůznějších druhů ať už menších lokálních nebo celosvětových, jako příklad, si lze například uvést ruskou V Kontakte, americkou filmovou databázi IMDb, aukční server eBay nebo českou verzi Aukro. Pokud považujeme za sociální sítě obecně za určité schéma, pomocí kterého zachycujeme vztahy mezi lidmi, pak můžeme k sociálním sítím zařadit i síť peněžních transakcí, viz například Bitcoinová síť. [8] Peněžní transakce vytvářejí mezi lidmi určité vztahy, které můžeme modelovat pomocí grafové databáze.

[7] Obr. 5 Zdroj: http://vincos.it/world-map-of-social-networks/



Obrázek 6 - Model bitcoinové sítě[8]

[8] Obr. 6 Zdroj: http://blog.bitcoinbeginner.com

6 Popis a implementace vybrané grafové databáze – Neo4j

Grafových databází existuje ve světě mnoho druhů, mezi nejpoužívanější patří například: Neo4j, Titan, OrientDB, InfiniteGraph, FlockDB a mnoho dalších. Po sturčné analýze dostupných databází byla však pro další práci vybrána databáze Neo4j. Důvody jsou následující:

- Databáze je open-source
- Má přehledně zpracovanou dokumentaci
- Od verze 2.0 je dostupné web rozhraní pro přístup na server
- Databáze je nestále vyvíjena
- Je nejpoužívanějším řešením grafových databází [9]

Neo4j je jedna z nejpopulárnějších grafových databází s open-source licencí implementovaná v jazyku Java. Vytvořena byla společností Neo Technology, a od vydání verze 1.0 v roce 2010 se stala populární databází s velkým zastoupením ve vývojářské komunitě. Na rozdíl od relačních databází a ukládání dat do tabulek, v Neo4j jsou všechna data uložena do grafové struktury obsahující uzly, hrany nebo atributy. Každý z uzlů a hran může mít množství atributů. [10]

Popis

Popišme si pár důležitých předností Neo4j databáze. Hlavní předností je rychlost databáze, díky rychlosti vyhledávání a provádění operací v databázi je Neo4j vhodné i pro real-time dotazování. Datové úložiště skladuje data v jejich přirozené podobě, to umožňuje databázi rychle se přizpůsobovat změnám, novým zdrojům dat a výjimkám v pravidlech. Databáze prochází agilním vývojem a rychle reaguje na nové požadavky. Na rozdíl od některých NoSQL databází, Neo4j využívá podporu ACID, čímž je zaručena konzistence dat v databázi. Další výhodou Neo4j je hezky zpracované grafické prostředí dostupné prostřednictvím webového prohlížeče. Skrz toto prostředí lze provádět dotazy prostřednictvím jazyků Cypher, Gremlin apod. a dále přistupovat k informacím o datech, vykreslování grafů a dalšímu. Cypher je deklarativní dotazovací jazyk pro Grafovou databázi Neo4j pro expresivní a efektivní dotazovaní a updatování grafového úložiště. Cypher je relativně jednoduchý, ale velmi schopný jazyk, a i velmi komplikované dotazy můžou být jednoduše vyjádřeny jazykem Cypheru. [11]

Díky populárnosti Neo4j je databáze kompatibilní s velkou základnou programovacích jazyků a frameworků. Možnostmi zadávání příkazů pro cypher je přímo integrované REST API nebo množství ovladačů pro dané jazyky. Další z výhod je snadné načtení dat z jiného druhu databáze pomocí zabudovaného Cypher LOAD CSV pluginu.

6.1 Dotazovací jazyky

Hlavním dotazovacím jazykem Neo4j je jazyk Cypher, avšak díky podpoře ostatních jazyků, existují i alternativy pro dotazování na databázi. Jednou z těchto alternativ je například traverzovací jazyk Gremlin.

- Cypher – je jazyk primárně určený pro práci s daty v databázi. Umožňuje efektivní dotazování a updatování dat. Svou strukturou je jazyk podobný struktuře SQL dotazů.
- Gremlin – jazyk nativně zaměřený na traverzování grafem. Pomocí příkazů zadaných do příkazového řádku/konzole lze interaktivně procházet jednotlivými částmi grafu.

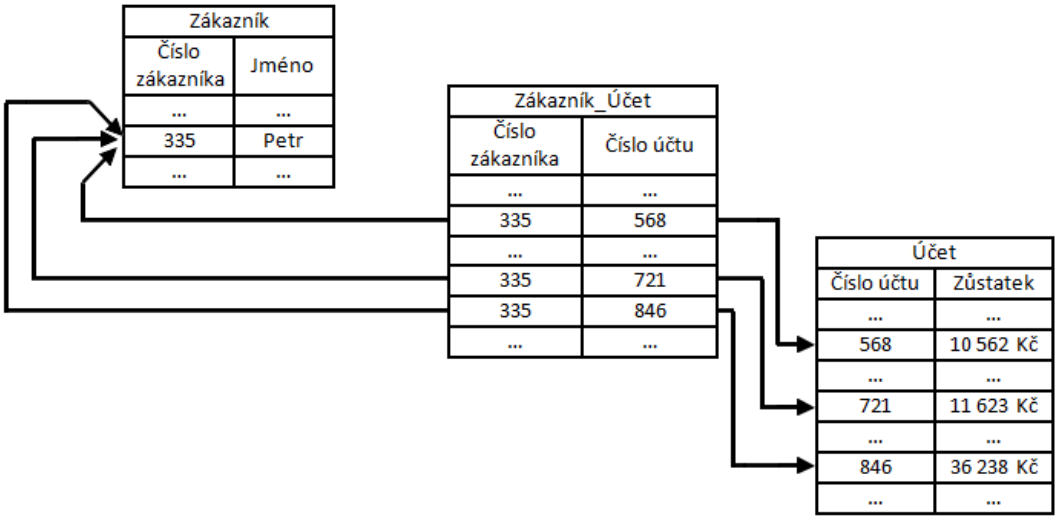
Traverzování neboli průchod nám umožňuje navštívit všechny vrcholy grafu. Celý proces si můžeme představit jako systematické putování grafem po jednotlivých vrcholech a hranách. Způsob, jakým je průchod realizován, určují zvolené algoritmy, jakými jsou například průchod do hloubky či do šířky. [12]

6.2 Porovnání modelu uložení dat relační databázi a grafové databázi

Pokud porovnáme způsob uložení dat v relačních a grafových databázích, uvidíme, že grafové databáze mohou být pro uživatele v určitých případech intuitivnější než relační databáze. Pokusme si proto na následujícím vzorovém příkladě popsat jak jsou data uložena a provázána v případě grafové a relační databáze.

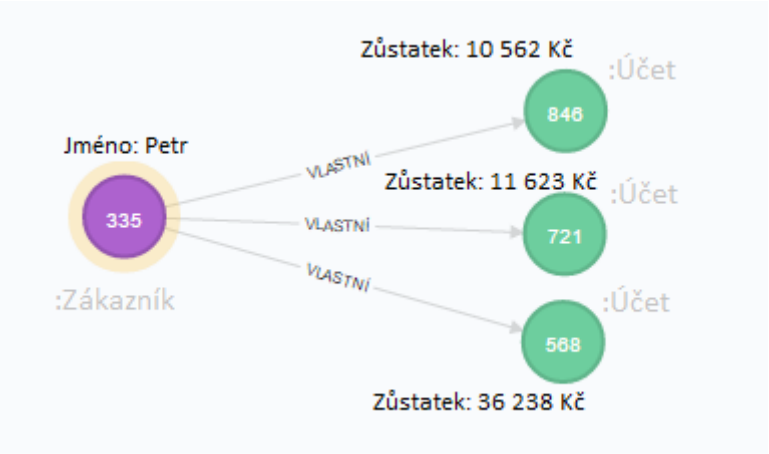
Za jednoduchý příklad může sloužit vlastnictví osobních účtů v bance. Petr vlastní v bance celkem 3 účty, každý z těchto účtů obsahuje určitý zůstatek na kontě. Pro náš příklad Petr vlastní svůj údaj, kterým je jeho uživatelské číslo, pomocí kterého je bankou identifikován.

Na obráku 7 lze vidět strukturu relační databáze



Obrázek 7 - Model uložení dat v Relační databázi

V případě relační databáze je třeba asociační tabulky, která uchovává data o vlastnictví daného účtu. U databáze grafové můžeme namodelovat uzel typu :Zákazník a :Účet obsahující atributy definující Petra a jeho účty. Mezi nimi bude hrana označena :Vlastní která nám propojuje účet s jeho majitelem.



Obrázek 8 - Model uložení dat v Grafové databázi

6.3 Datový model

Grafová databáze Neo4j používá „bezschémový“ model, to znamená, že jeho použití není nutné. Data ukládaná do databáze mohou být modelována tak, aby odpovídala co nejvíc realitě. Data uložená formou již výše zmíněných grafů obsahujících uzly, hrany a atributy vytvářejí tzv. labeled property graph model. Tento výsledný model obsahuje cesty (paths). Cesta je jeden nebo více uzlů spojených vztahy. Nejmenší možná cesta je délky 0, tj. jeden samotný uzel.

Důležitým prostředkem pro práci s grafem jsou vzory (patterns). Ty jsou využívány například k definování dotazů pro jazyk Cypher. Základním vzorem je jednoduchý uzel „(n)“, kde n nám identifikuje uzel. Identifikace uzlů není nezbytná, používá se však pro pozdější použití uzlu. Pro vytváření složitějších vzorů je důležité znázornění vztahů mezi uzly. Vztahy jsou znázorněny pomocí „-->“ nebo „--“ v případě, že není definován směr vztahu. Pro vztah mezi dvěma uzly je tedy použita notace „(n)-->(m)“.

Další důležitou součástí vzorů je využívání Labelů nebo také štítků. Label slouží k rozdělení uzlů do menších množin podle stejných Labelů. Jeden uzel může obsahovat i více labelů. V případě, že uzel je opatřen štítkem používáme označení např. „(m:Movie)“, tímto labelem lze definovat, že daný uzel patří mezi filmy.

Atributy uzlů jsou vyjádřeny pomocí složených závorek, kde jednotlivé páry typu key-value jsou odděleny čárkou. Např.: „(m:Movie {title: „The Matrix“, released: 1999}).

Stejným způsobem definujeme labely a atributy i pro vztahy: „(n)-[SENT{count: 10}]-(m)“.

Pro případy traverzování grafem jsou definovány vzory: „(n)-[*2]->(m)” a „(n)-->()-->(m)” jsou ekvivalentní, oba znázorňují cestu délky 2. Pokud je třeba znázornit cestu intervaly, lze

použít například vzor „(n)-[*2..6]->(m)" odpovídá délce cesty nejméně 2 ale maximálně 6. Při použití vzoru bez definování délky „(n)-[*]->(m)" nalezneme všechny cesty, avšak tento vzor není vždy výhodný vzhledem k velikosti výsledného grafu.

6.4 Cypher

Dotazování na graf probíhá pomocí příkazů, představme si proto základní příkazy.

Příkazy umožňující získání potřebných dat z databáze.

- Match – hledá odpovídající vzor dotazu v datech
- Where – slouží k filtrování dle zadaných hodnot
- Return – navrací a zobrazuje výsledná data
- Order by – třídí výsledek dotazu podle zadaného argumentu
- Skip/Limit – upravuje rozsah dat, která jsou prohledávána

Příkazy umožňují vytvářet strukturu databáze.

- Create – vytváří uzly a hrany
- Merge – vytváří unikátní uzly
- Create Unique – vytváří unikátní hrany
- Delete – maže uzly a hrany
- Set – upravuje vlastnosti a labely
- Remove – odstraňuje vlastnosti a labely

Podobně jako u SQL i Cypher podporuje agregační funkce.

Agregační funkce jsou funkce, pomocí kterých lze seskupit vybrané řádky dotazu a spočítat nad nimi výsledek určité aritmetické nebo statistické funkce.

- Count – součet čísel
- Min /max – najde nejmenší/největší hodnotu
- Avg – průměr hodnot
- Collect – kolekce hodnot z dat

Příklady použití:

```
CREATE (m:Movie {title:"The Matrix", released:1999});
```

Prostřednictvím příkazu CREATE vytvoříme uzel označený Labelem :Movie. Tento uzel obsahuje dva atributy nazvané „title" a „released". K tomuto účelu lze použít i příkaz MERGE, který zároveň zabraňuje duplicitnímu vytvoření uzlu.

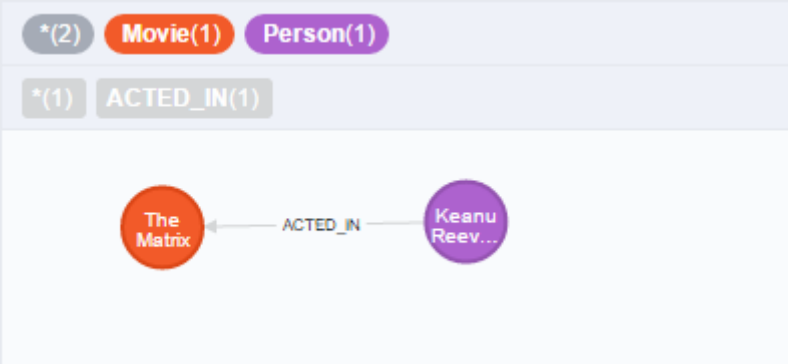
Dále považujme za vytvořený uzel s Labelem :Person a atributy „name" a „born".

```
(p:Person {name:"Keanu Reeves", born:1964});
```

Mezi tyto uzly vytvoříme hranu -[:ACTED_IN]-> příkazem

```
MATCH (m:Movie {title:"The Matrix"}),
(p:Person {name:"Keanu Reeves"})

MERGE (p)-[r:ACTED_IN{role:"Neo"}]->(m);
```

Výsledek vidíte na obr. 9.



Obrázek 9 - Graf v Neo4j

Obdobným způsobem lze použít příkaz DELETE ke smazání jednotlivých uzlů, části nebo celého grafu.

```
MATCH (m:Movie {title:"The Matrix"})

RETURN m;
```

pro oba uzly a hranu použijeme následující příkaz

```
MATCH (m:Movie {title:"The Matrix"})<-[r:ACTED_IN]-(p:Person
{name:"Keanu Reeves"})

DELETE m,p,r;
```

Jak lze vidět v dotazech je používán příkaz MATCH pro nalezení odpovídajících uzlů či hran v grafu, prostřednictvím příkazu RETURN lze definovat části grafu, které mají být vypsány či zobrazeny.

6.5 Instalace a systémové požadavky serveru

V případě námi používaných Windows 8.1 je instalace prováděna pomocí instalačního souboru dostupného na stránkách Neo4j. V době instalace balíčku byla dostupná verze 2.2.0.

Neo4j Community běží v našem případě jako server dostupný pomocí web rozhraní, REST rozhraní nebo prostřednictvím ovladačů pro specifické programovací jazyky.

Požadavky:[9]

	Minimální	Doporučená
Procesor	Intel Core i3	Intel Core i7
Paměť	2GB	16-32GB
Disk	10GB SATA	SSD SATA
Souborový systém	Ext4 (nebo podobný)	Ext4
Operační systém	Windows, Mac OS X, Linux	Windows, Mac OS X, Linux

V našem případě server běží na konfiguraci:

Procesor: Intel Core i5

Paměť: 4GB

Disk: 1TB HDD

Souborový systém: NTFS

Operační systém: WIN8.1 Pro

6.6 Vizualizační nástroje Neo4j

6.6.1 Neo4j WebUI

Důležitou částí Neo4j Community Serveru je i Web User Interface připravené pro manipulaci s daty v databázi a jejich vizualizaci. Prostřednictvím tohoto rozhraní lze provádět dotazy pomocí jazyka Cypher. Defaultně je dostupná po spuštění Neo4j Serveru na adrese http://localhost:7474/

6.6.2 Browser

Dotazy vkládáme do editoru, po úspěšném provedení dotazu přejde dotaz do Streamu, kde se zobrazí jako graf nebo v tabulkové podobě. Výsledky dotazů mohou být snadno exportována do různých formátů jako CSV, JSON, PNG apod.

V případě jednořádkových dotazů lze potvrdit dotaz pomocí tlačítka <enter> nebo pomocí ikony ve tvaru „šipky“, pro větší přehlednost delších dotazů lze pomocí <shift-enter> přepnout editor do módu „multi-line“ v tomto případě lze dotaz stylizovat do více řádků za účelem zachování přehlednosti prováděného dotazu. Vykonání dotazu lze provést pomocí zkratky <ctrl-enter> což následuje zobrazení dotazu ve streamu.

Historií dotazů lze listovat pomocí šipek nahoru a dolů pro poslední dotazy a v případě delší historie lze použít zkratku <ctrl> + šipka nahoru/dolů.

[9] Zdroj: http://neo4j.com/docs/2.2.1/deployment-requirements.html

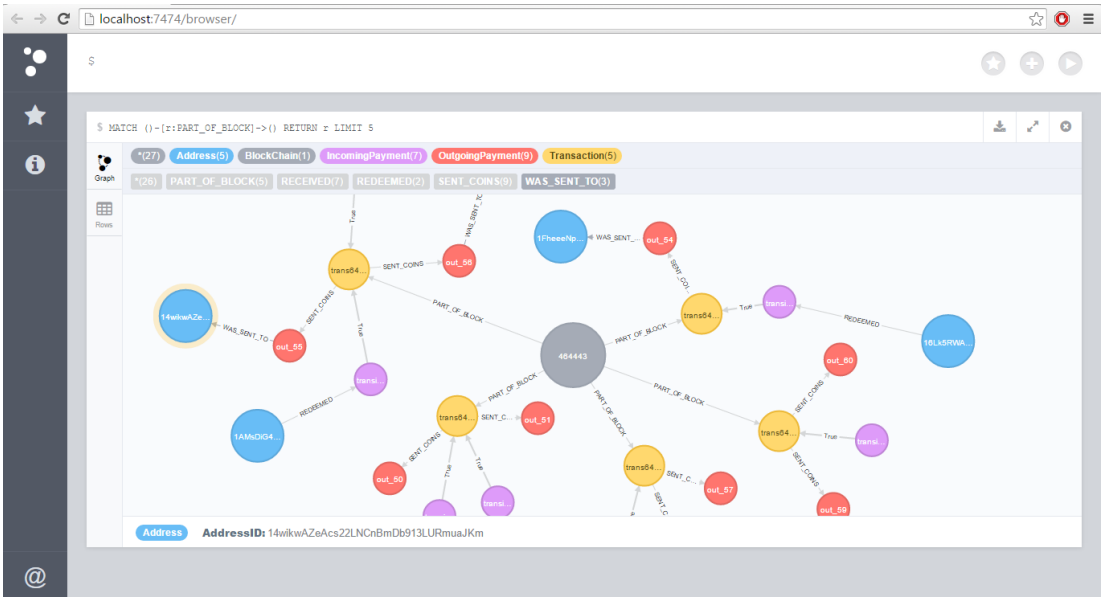
Další zajímavou možností Browseru je možnost uchovávání dotazů v editoru pro pozdější použití pomocí ikony se symbolem „hvězdy“. Poté co jsou data uložena, jsou dostupná pomocí boční lišty, kterou si popíšeme níže. V případě že bude první řádka dotazu komentářem, poslouží při uložení jako název dotazu.

Boční lišta slouží ke snadnější orientaci v datech uložených v databázi, usnadnění používání častých skriptů jejich uložením mezi oblíbené. A umožňuje nám vstup do dokumentace a pomocných informací o Neo4j. Také umožňuje vstup do WebAdmin sekce.

První záložkou v boční liště se dostaneme do přehledu databáze, tato záložka nám umožňuje snadnější orientování v dostupných Labelech vrcholů, typů vztahů a atributů objevujících se v databázi.

Druhá záložka slouží pro zobrazení uložených dotazů, možnosti přizpůsobení grafového zobrazení. Poslední možností této záložky je přímé vložení datového souboru obsahujícího skript nebo import malého množství dat.

Prostřednictvím třetí záložky lze přistupovat k dokumentaci Neo4j, základním příručkám, ukázkovým příkladům a grafům, pomocným materiálům a do WebAdmin části.



Obrázek 10 - Neo4j Browser

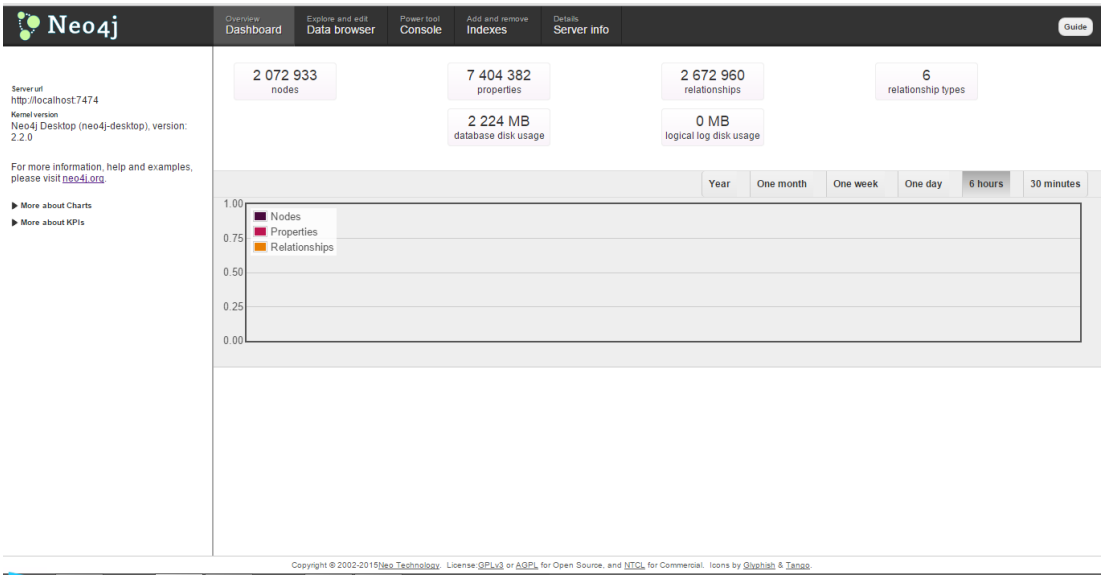
6.6.3 WebAdmin

WebAdmin interface je dostupný na adrese localhostu stejně jako základní browser. Defaultní adresou WebAdminu je http://localhost:7474/webadmin/.

Základními prvky WebAdminu jsou záložky Dashboard, Data Browser, Console, Indexy a Server Info. Prostřednictvím těchto záložek lze přistupovat k pokročilejším vlastnostem Neo4j Serveru, vizualizovat a měnit data, sledovat změny počtu dat v databázi apod. Představme si postupně jednotlivé z těchto částí podrobněji.

6.6.3.1 Dashboard

Na první pohled poskytuje dashboard monitorování obsahu databáze a její aktivity v čase. Ačkoliv jsou hodnoty dat poskytovány pouze jako odhady počtu uzlů, vztahů a atributů místo přesných čísel, může dashboard sloužit pro udělání si obrázku o velikosti a struktuře databáze a ne pro přesné měření dat.

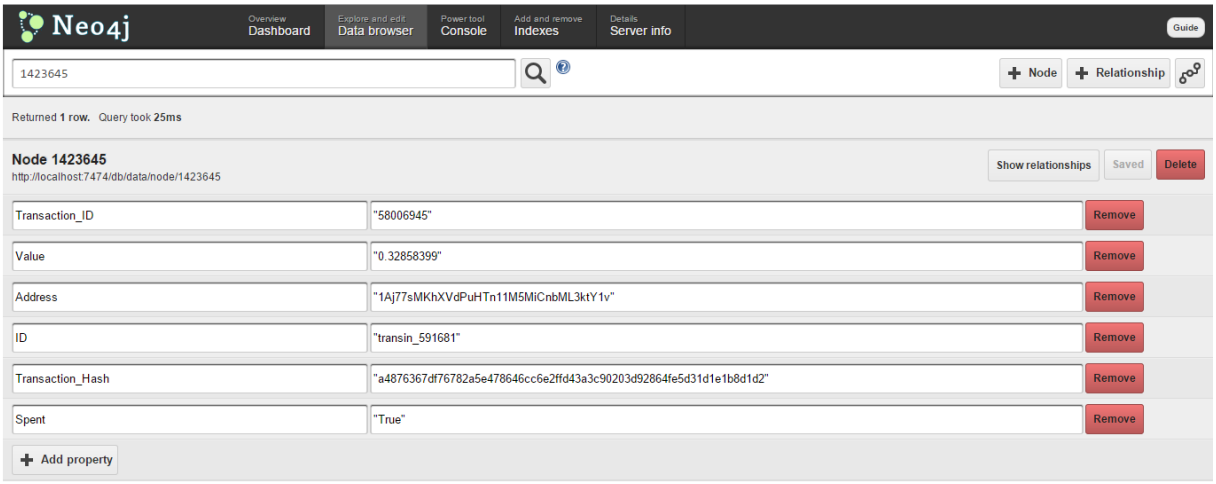


Obrázek 11 - Neo4j Dashboard

6.6.3.2 Data Browser

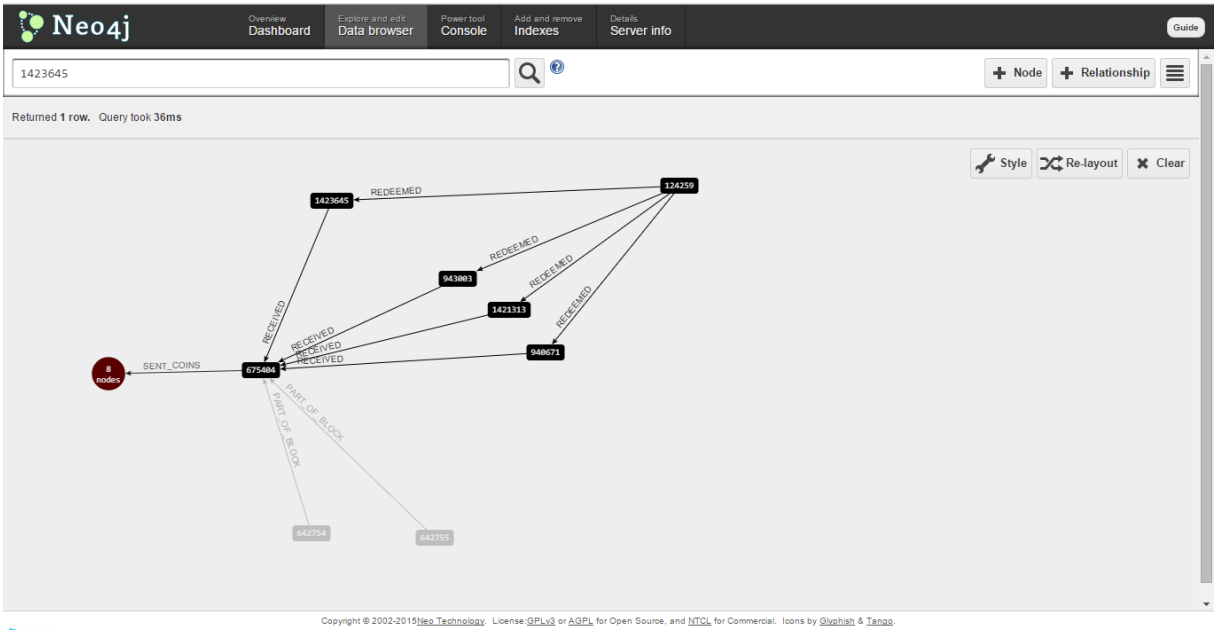
Data Browser slouží k interpretaci dat uložených v databázi. Interpretování dat může probíhat dvěma způsoby nazvanými inspect a visualize.

Inspection view je první formou, zobrazuje detaily o uzlech, vztazích a hodnotách atributů. Prostřednictvím formuláře lze uzly, vztahy i atributy přidávat, měnit jejich hodnotu či mazat. Dotazování může probíhat pomocí přímého dotazu na uzel, vztah nebo například pomocí Cypher dotazů. Inspection view je vhodný pro jednoduché vytváření dat a pracování s nimi.



Obrázek 12 - Neo4j Data browser, Inspection

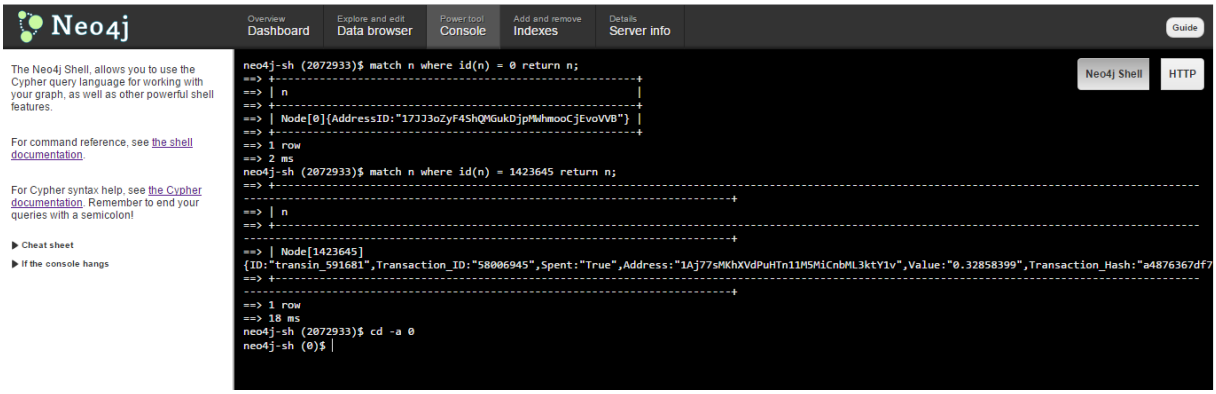
Druhou formou je Vizualization view, který nám umožňuje zobrazit data z inspection view ve formě grafu. Kliknutím na ikonku  dojde k přepnutí zobrazení a umožní nám to ukázat cesty grafu. Kliknutím na šedivé uzly lze graf interaktivně rozbalovat a procházet. Stejně jako graf ve streamu Browseru může být i tento stylizován do podoby, kterou požadujeme.



Obrázek 13 - Neo4j Data Browser, Virtualize

6.6.3.3 Console

Console je silným nástrojem webadmin rozhraní, pomocí ní lze zadávat příkazy Cypheru nebo testovat server pomocí REST API.[10]



Obrázek 14 - Neo4j Console

[10] REST – Representional State Transfer- slouží k vytvoření, čtení, editování nebo mazání informací ze serveru pomocí http volání.

6.6.3.4 Server Info

Sekce server info obsahuje informace o databázi a serveru.

6.6.3.5 Shrnutí

Prostřednictvím web rozhraní lze získat plnou kontrolu nad všemi částmi Neo4j. Ať už se jedná o jednodušší rozhraní browseru, či pokročilejší rozhraní webadmin. Tato rozhraní se s každou verzí vyvíjí a přinášejí nové možnosti.

7 Analýza dat pomocí grafové databáze

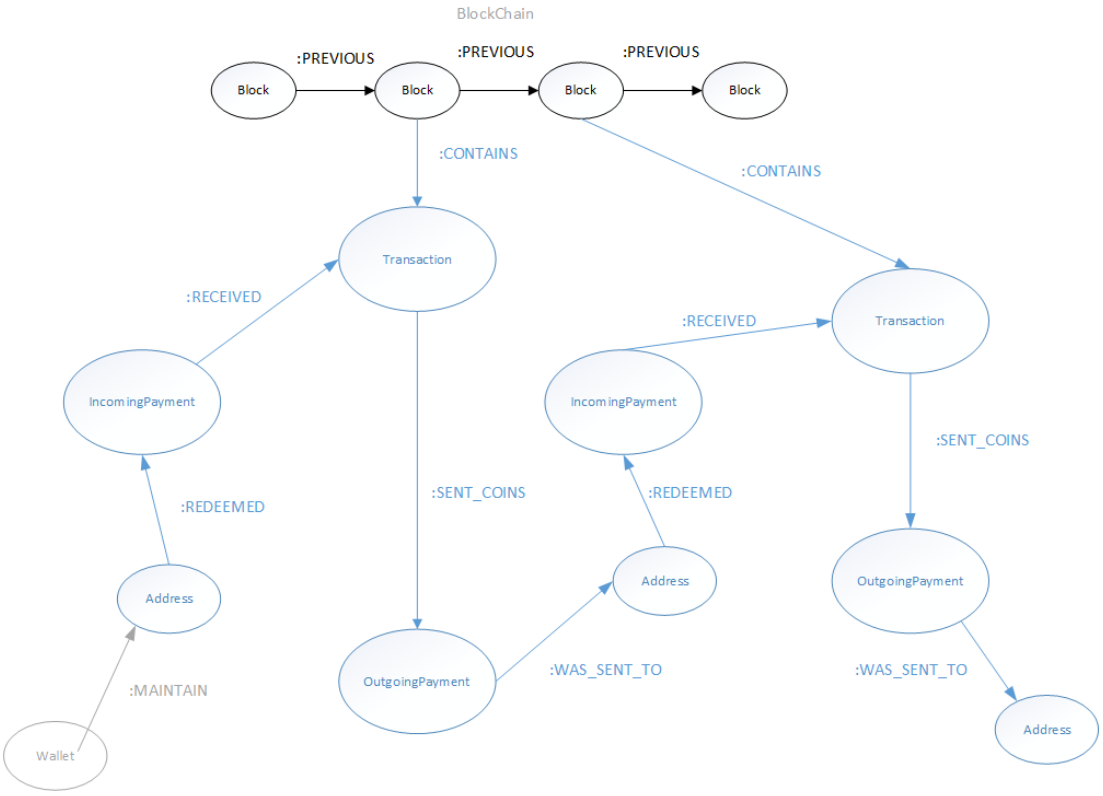
Analýzu dat pomocí grafové databáze provádíme na datech z transakcí pomocí digitální měny Bitcoin.

7.1 Blockchain.info

Blockchain.info je populární bitcoinová peněženka spuštěná roku 2011 Benem Reevesem, vyvíjená ve Velké Británii. Blockchain není přímo sociální sítí, ale svou strukturou jí připomíná. Služba poskytuje data o transakcích, „vydolovaných" blocích, statistiky o bitcoinové ekonomice a prostředky pro developery. Roku 2013 byla nejvíce navštěvovanou bitcoin stránkou na světě s více než 118 miliony návštěv. [13]

7.2 Import

Prostřednictvím postupu ze serveru Github [14] jsme získali část dat ze serveru Blockchain.info naše data jsou uložena na disku ve formátu CSV s hlavičkami. Importování probíhalo pomocí console a Cypheru prostřednictvím LOAD CSV dotazu. CSV soubory byly uloženy na disku a jejich velikost byla přibližně 400MB. Z dat získaných z Blockchain.info došlo i k navrhnutí předpokládané struktury databáze, která obsahuje celkem 5 typů uzlů a 6 typů vztahů. Navržený model dat lze vidět na obr. 15. Pro lepší přehlednost, se v navrženém modelu nachází i uzel Wallet, pro přesnější znázornění struktury bitcoin sítě.



Obrázek 15 - Model Blockchain databáze

Prostřednictvím následující sekvence příkazů Cypheru byl proveden import dat do databáze.

Cypher:

```
create index on :Address(AddressID);

create index on :BlockChain(ID);

create index on :Transaction(ID);

create index on :IncomingPayment(ID);

create index on :OutgoingPayment(ID);


//poc_adresslist

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_addressList.txt" as line

CREATE (:Address {AddressID:line.AddressID});


//poc_blockdata

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_blockdata.txt" as line

CREATE (:BlockChain {ID:line.ID, BlockID:lineBlockID, Hash:
line.Hash, Received_Time:line.Received_Time,
Previous_Block_Hash:line.Previous_Block_Hash,
Transaction_Count:line.Transaction_Count,
Height:line.Height});


//poc_transaction

USING PERIODIC COMMIT 10000
```

```
LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_transaction.txt" as line

CREATE (:Transaction {ID:line.ID,
Transaction_ID:line.Transaction_ID, Hash:line.Hash,
Time:line.Time, V_In:line.V_In, V_Out:line.V_Out});


//poc_transactionsIn

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_transactionsIn.txt" as line

CREATE (:IncomingPayment
{ID:line.ID,Transaction_ID:line.Transaction_ID,
Transaction_Hash:line.Transaction_Hash, Address:line.Address,
Spent:line.Spent, Value:line.Value});


//poc_transactionsOut

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_transactionsOut.txt" as line

CREATE (:OutgoingPayment {ID:line.ID,
Transaction_ID:line.Transaction_ID,
Transaction_Hash:line.Transaction_Hash, Type:line.Type});



//poc_relsBlocksTransactions

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_relsBlocksTransactions.txt" as line

MATCH (b:BlockChain {ID: line.START_ID})
```

```
MATCH (t:Transaction {ID:line.END_ID})

CREATE (b)-[:PART_OF_BLOCK]->(t);


//poc_relsPaymentsToAddress

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_relsPaymentsToAddress.txt" as line

MATCH (o:OutgoingPayment {ID: line.START_ID})

MATCH (a:Address {AddressID:line.END_ID})

CREATE (o)-[:WAS_SENT_TO{Spent:line.Spent, Value:line.Value}]-
>(a);


//poc_relsRedeemedFRomAddress

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_relsRedeemedFromAddress.txt" as line

MATCH (a:Address {AddressID:line.START_ID})

MATCH (i:IncomingPayment {ID:line.END_ID})

CREATE (a)-[:REDEEMED{Spent:line.Spent, Value:line.Value}]-
>(i);


//poc_relsTransactionsIn

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_relsTransactionsIn.txt" as line

MATCH (i:IncomingPayment {ID:line.START_ID})

MATCH (t:Transaction {ID:line.END_ID})
```

```
CREATE (i)-[:RECEIVED{Spent:line.Spent, Value:line.Value}]-
>(t);


//poc_relsTransactionsOut

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_relsTransactionsOut.txt" as line

MATCH (t:Transaction {ID:line.START_ID})

MATCH (o:OutgoingPayment {ID:line.END_ID})

CREATE (t)-[:SENT_COINS{Spent:line.Spent, Value:line.Value}]-
>(o);


//previous

USING PERIODIC COMMIT 10000

LOAD CSV WITH HEADERS FROM
"file:///D:/Skola/Vejska/!Bakalarska%20Prace/Prak.%20data/bitc
oin/data-kopie/poc_blockdata.txt" as line

MATCH (a:BlockChain {ID:line.ID})

MATCH (b:BlockChain {ID:line.Previous_Block_Hash})

CREATE (a)-[:PREVIOUS]->(b);
```

Po provedení importu měla databáze konečnou podobu s přibližně 2 miliony uzlů, 2,6 milionu vztahů a 7,4milionu atributů.

8 Využití grafové databáze pro analýzu

Analýza, kterou provádíme je zaměřena na platby zaplacené pro výkupné viru Cryptowall. Cryptowall je ransomwarový virus, který napadá soubory našeho počítače, a všechny soubory jím napadeny jsou zašifrovány s pomocí šifrování RSA 2048. Po zašifrování souborů je jednou z možností získat zpět svá data pomocí zaplacení výkupného. Naše platba probíhá anonymně pomocí sítě Tor a požadovanou částku 500$ je potřeba uhradit v bitcoinech. Díky této metodě se daří držet hackerům, kteří vytvořili virus Cryptowall v tajnosti.

8.1 Prvky podrobené analýze

- **Address**

Bitcoin adresa je udávaná stringem o počtu 26-35 znaků ve tvaru například "1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV". Adresa může být vygenerována pro každého uživatele Bitcoinů. Bitcoinová adresa může být svým způsobem přirovnána k emailové adrese, pokud chce uživatel zaslat bitcoiny jinému uživateli, pošle je na jeho adresu.

- **BlockChain**

Blockchain je sdílený veřejný log transakcí. Každý bod bitcoin sítě tento log obsahuje. V blockchainu jsou obsaženy všechny uskutečněné transakce a díky tomu se lze dozvědět, která adresa obsahuje kolik bitcoinů. Díky tomu dokáže Bitcoin peněženka spočítat aktuální stav bitcoinů a nová transakce může být provedena po ověření dostupných prostředků. Jak již název napovídá, Blockchain je „řetěz" menších jednotek nazvaných „Block". Každý z bloků obsahuje hash předchozího, proto jsou všechny bloky řazeny chronologicky po sobě od původního bloku až k poslednímu.

- **Block**

Každý z bloků obsahuje potvrzené a nepotvrzené (čekající) transakce za určitý časový úsek. Prostřednictvím bloků lze sledovat směry plateb mezi adresami. Po uběhnutí časového úseku dojde k pernamentnímu zaznamenání dat zapsáním Blocku do Blockchainu.

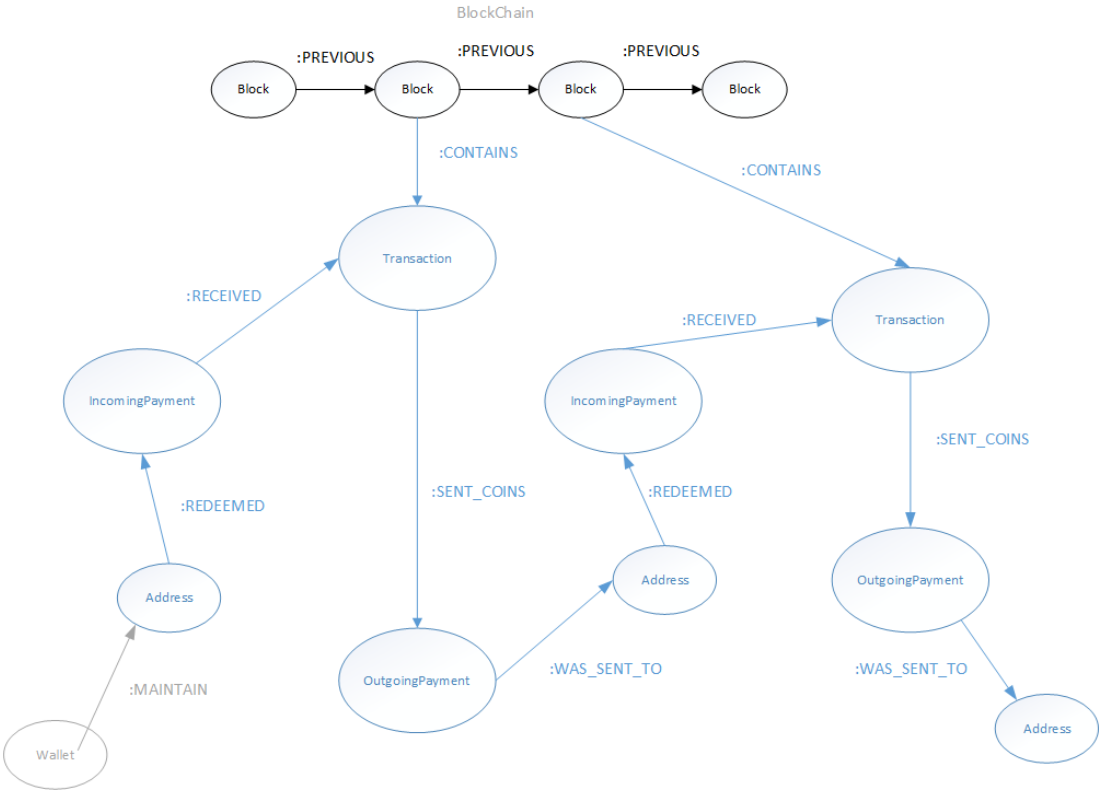
- **Transaction**

Transakce je převod bitcoinů mezi adresami, který je součástí blockchainu a je přenášen bitcoin sítí.

- **Wallet**

Wallet neboli „peněženka" slouží k přehledu financí, prostřednictvím peněženky lze přistupovat k vlastním bitcoinům a adresám. Pomocí peněženky lze také poslat bitcoiny jiným uživatelům, pokud známe jejich adresu.

Jako první krok naší analýzy si porovnáme naše schéma se schématem reálně obsaženým v databázi. Schéma navržené pro importaci dat vypadá následovně:



Obrázek 16 - Model Blockchain databáze

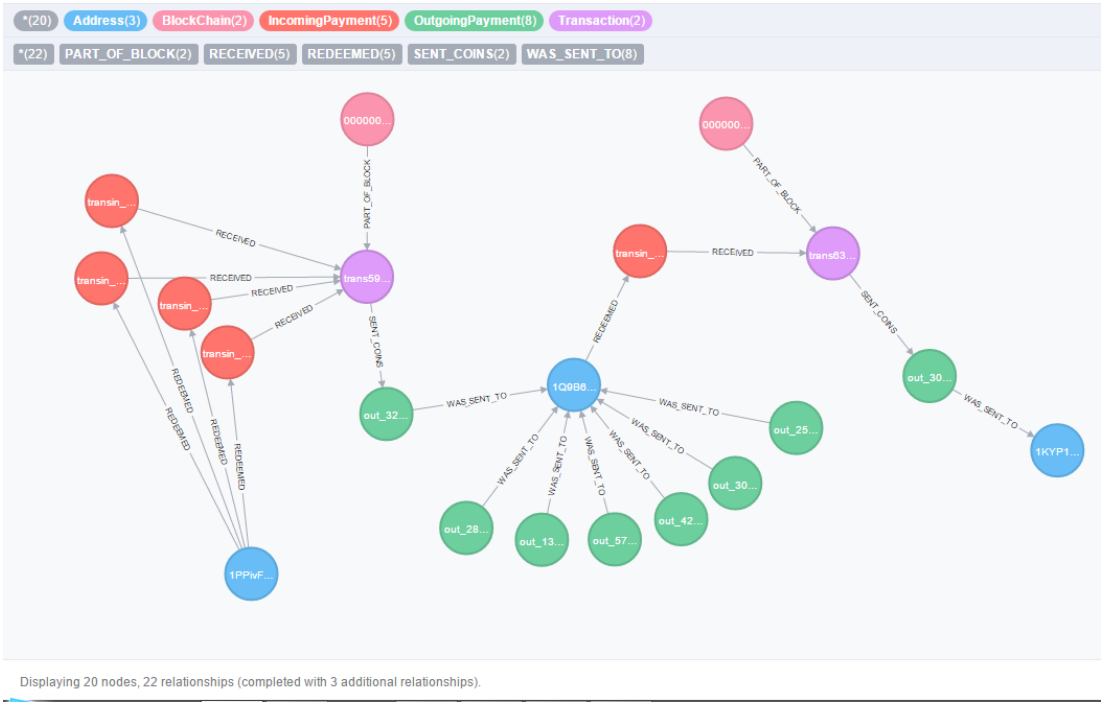
Ve chvíli kdy dojde k vyvolání platby peněženkou, odešle se částka prostřednictvím uzlu IncomingPayment do uzlu Transaction obsahujícího údaje o transakci a následně dojde k odeslání financí přes uzel OutgoingPayment do cílového uzlu Address. Jak je z grafu vidět, každá z transakcí je vázána na Block v Blockchainu.

Pro znázornění obdobné situace použijeme Neo4j Browser, kde prostřednictvím editoru a cypher dotazu najdeme jednu náhodnou adresu a po rozbalení jejích cest se traverzováním přiblížíme našemu předpokládanému grafu.

Cypher:

```
MATCH (n:Address)

WHERE n.AddressID="1Q9B6g3cQ2qrYRNyYQbSrsRRmzQr7fMWQd"

RETURN n;
```

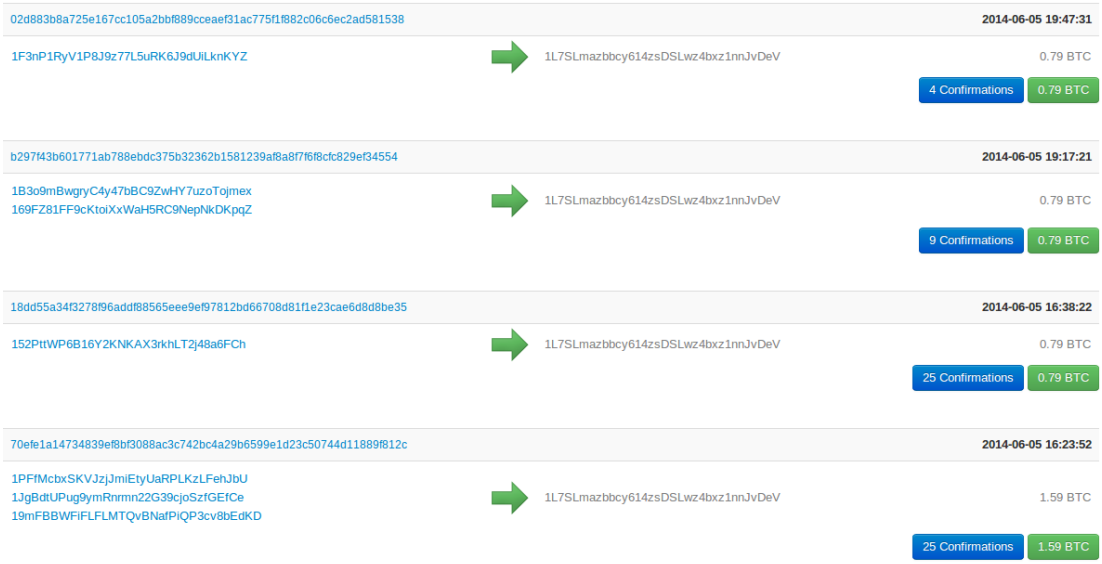
Po rozbalení cest z uzlu Address jsme zobrazili výsledný graf, který svou strukturou odpovídá původně navrhnutému modelu databáze.



Obrázek 17 - Porovnání předpokládané struktury s reálnou strukturou

V článku publikovaném 4. Listopadu 2014 na Blogu Cylance [15] se autor zabýval problematikou viru Cryptowall a prostřednictvím vlastních metod došel k vypátrání Bitcoinových adres používaných pro převody peněz získaných z výkupného. Proto se i my zaměřme na tyto adresy, abychom zjistili, zdali jsou stále využívány a jestli jsou spolu nějak propojeny.

Prostřednictvím bitcoin sítě autoři článku sledovali transakce přibližné hodnoty 0,79BTC a 1,59BTC, to jsou částky odpovídající k datu 5. Června 2014 přibližně částkám 500$ a 1000$ požadovaným virem Cryptowall pro zaplacení výkupného. Na obrázku níže lze vidět několik provedených plateb na Bitcoin adresu „`1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV`“, proto se lze domnívat, že je jednou z adres používaných pro přijímání plateb.



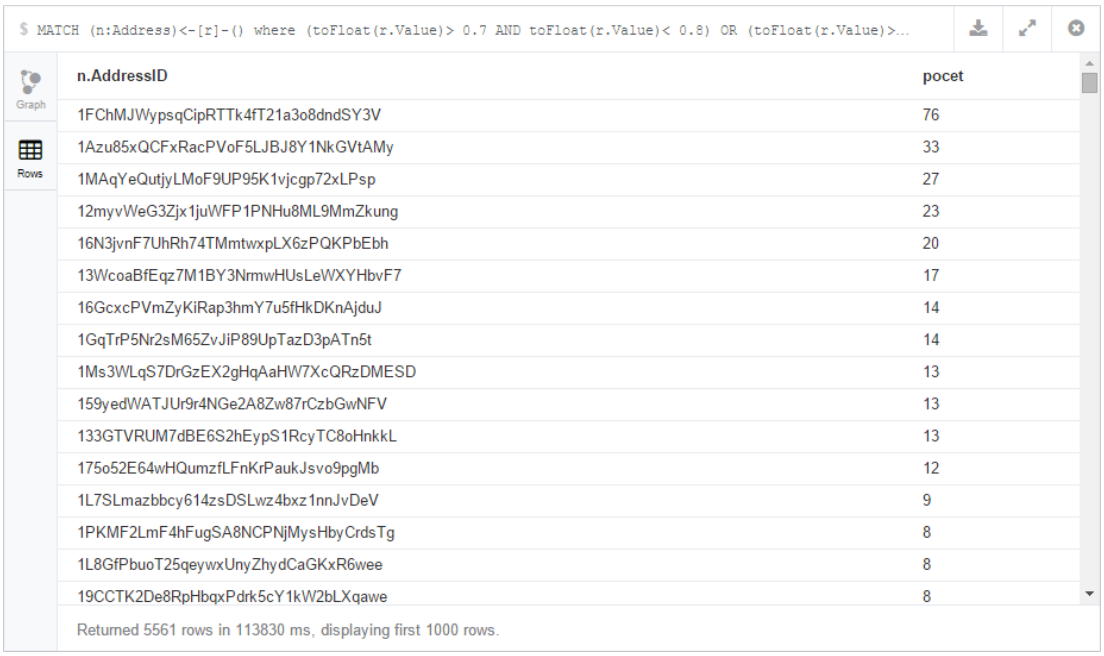
Obrázek 18 - Pozorování bitcoin transakcí[11]

Díky poznatkům ze článku se proto můžeme zaměřit právě na adresy podobné, proto pomocí následujícího Cypher dotazu prohledáme naší databázi a identifikujeme adresy přijímající podobné platby ve výši 0,79BTC a 1,59BTC, pro dotaz bylo nastaveno rozmezí 0,7-0,8BTC a 1,5-1,6BTC

Cypher:

```
MATCH (n:Address)<-[r]-()

Where (toFloat(r.Value)> 0.7 AND toFloat(r.Value)< 0.8) OR
(toFloat(r.Value)> 1.5 AND toFloat(r.Value)< 1.6)

RETURN n.AddressID,count(r) as pocet

ORDER BY pocet DESC;
```

[11] Obr.18 Zdroj: http://cdn2.hubspot.net/hub/270968/file-963913716-png/bots/cryptowall/Screenshot_-_06052014_-_012544_PM.png?t=1428429067623



Obrázek 19 - Počet podezřelých plateb pro BTC adresy

Díky výsledkům dotazu se nám povedlo zobrazit několik adres, jenž převyšující počtem plateb ostatní. Mezi tyto adresy se dostala naše původní adresa „1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV“ jak lze vidět na obrázku. Vyberme proto z tabulky pár adres a zaměřme se, co o nich lze zjistit.

Adresy:

- 1FChMJWypsqCipRTTk4fT21a3o8dndSY3V
- 1Azu85xQCFxRacPVoF5LJBJ8Y1NkGVtAMy
- 1MAqYeQutjyLMoF9UP95K1vjcgp72xLPsp
- 12myvWeG3Zjx1juWFP1PNHu8ML9MmZkung
- 16N3jvnF7UhRh74TMmtwxpLX6zPQKPbEbh
- 1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV

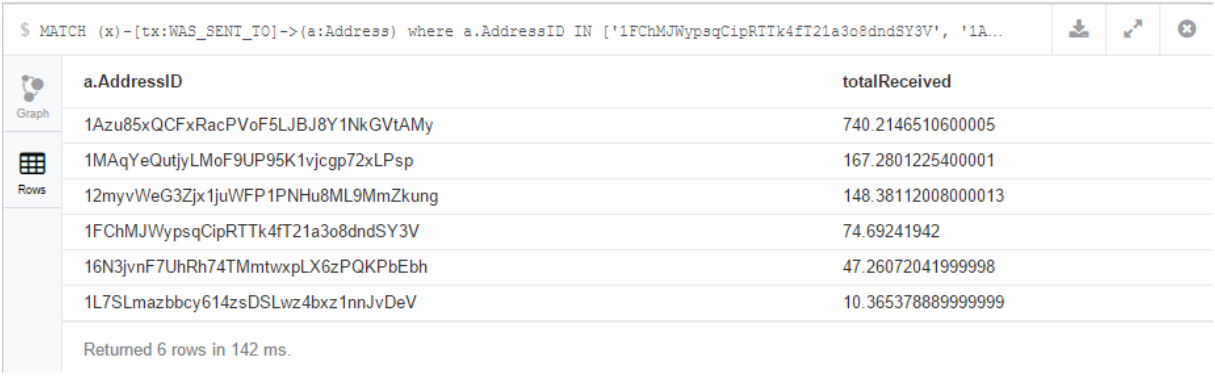
Následujícím dotazem jsme zjistili, kolik v naší databázi tyto adresy nashromáždily financí.

Cypher:

```
MATCH (x)-[tx:WAS_SENT_TO]->(a:Address)

where a.AddressID IN

['1FChMJWypsqCipRTTk4fT21a3o8dndSY3V',
'1Azu85xQCFxRacPVoF5LJBJ8Y1NkGVtAMy',
'1MAqYeQutjyLMoF9UP95K1vjcgp72xLPsp',
'12myvWeG3Zjx1juWFP1PNHu8ML9MmZkung',
```

```
'16N3jvnF7UhRh74TMmtwxpLX6zPQKPbEbh',
'1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV']

RETURN a.AddressID, sum(toFloat(tx.Value)) as totalReceived

ORDER BY totalReceived DESC;
```

Z výsledku dotazu lze usoudit, že hlavní příjem financí probíhá přes adresy '1Azu85xQCFxRacPVoF5LJBJ8Y1NkGVtAMy' a '1MAqYeQutjyLMoF9UP95K1vjcgp72xLPsp'.



Obrázek 20 - Získané finance celkem

I při aktuálně nižším kurzu než v roce 2014, je celková částka na našich sledovaných účtech dohromady 1188,194412 bitcoinu, po převodu do aktuálního kurzu[12] přibližně 295 028,6726USD (7 427 700,867CZK).

Pomocí následujících adres se pokusíme určit další na které jsou tyto adresy navázány. Pomocí cypher dotazu určíme, zdali alespoň dvě z našich předchozích adres zaslaly peníze na některou další adresu.

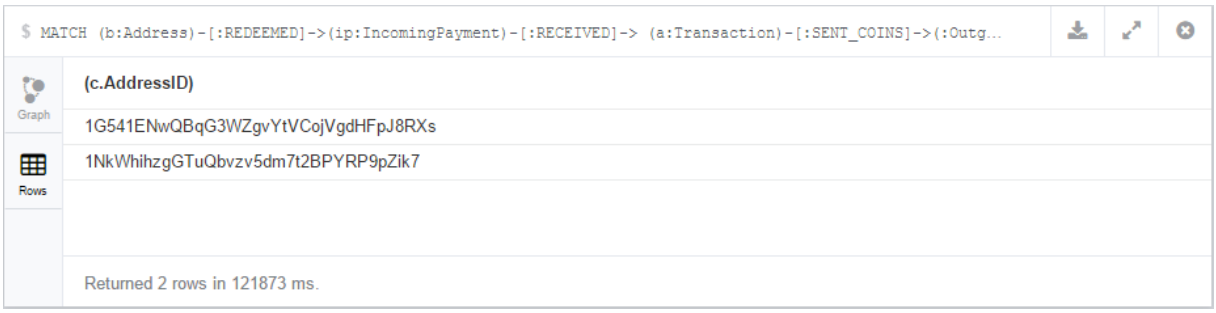
Cypher:

```
MATCH (b:Address)-[:REDEEMED]->(ip:IncomingPayment)-
[:RECEIVED]-> (a:Transaction)-[:SENT_COINS]-
>(:OutgoingPayment)-[:WAS_SENT_TO]->(c:Address),
(c:Address)<-[:WAS_SENT_TO]-(op:OutgoingPayment)<-
[:SENT_COINS]-(at:Transaction)<-[:RECEIVED]-
(ip2:IncomingPayment)<-[:REDEEMED]-(ba:Address)

WHERE b.AddressID IN ['1FChMJWypsqCipRTTk4fT21a3o8dndSY3V',
'1Azu85xQCFxRacPVoF5LJBJ8Y1NkGVtAMy',
'1MAqYeQutjyLMoF9UP95K1vjcgp72xLPsp',
'12myvWeG3Zjx1juWFP1PNHu8ML9MmZkung',
'16N3jvnF7UhRh74TMmtwxpLX6zPQKPbEbh',
'1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV']
```

[12] Dne 8.4.2015 – kurz 1 bitcoin = 248,3USD, 1 USD = 25,1762CZK

```
AND ba.AddressID IN ['1FChMJWypsqCipRTTk4fT21a3o8dndSY3V',
'1Azu85xQCFxRacPVoF5LJBJ8Y1NkGVtAMy',
'1MAqYeQutjyLMoF9UP95K1vjcgp72xLPsp',
'12myvWeG3Zjx1juWFP1PNHu8ML9MmZkung',
'16N3jvnF7UhRh74TMmtwxpLX6zPQKPbEbh',
'1L7SLmazbbcy614zsDSLwz4bxz1nnJvDeV']

AND b.AddressID <> ba.AddressID

RETURN distinct(c.AddressID);
```

Dvakrát bylo z našich adres zasláno pouze na dvě další adresy, díky tomuto dotazu se lze posouvat grafem dále a hledat další adresy navázané do sítě financí Cryptowallu.



Obrázek 21 - Adresy s vazbou na sledované adresy

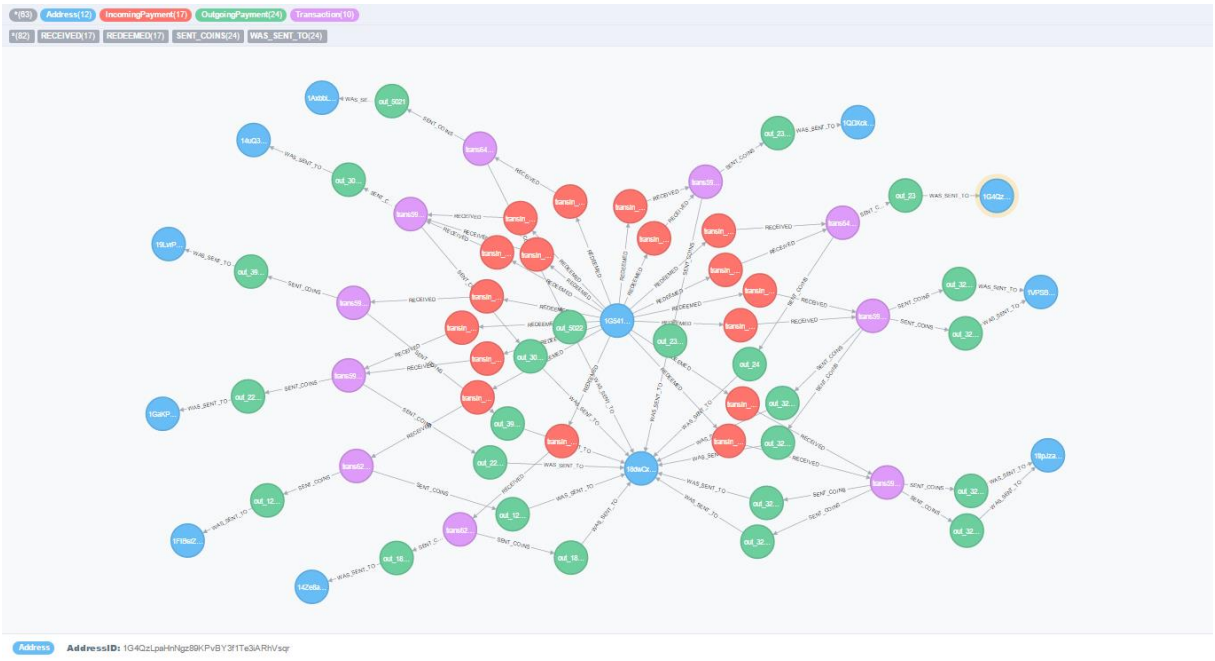
Provedení dotazu na výdělek jsme zjistili, že na tyto dvě adresy bylo odvedeno dohromady 72,61BTC.

Rychlou kontrolou lehce pozměněným dotazem na přijaté bitcoiny jsme zjistili, že z adresy „1G541ENwQBqG3WZgvYtVCojVgdHFpJ8RXs“ bylo obratem rozesláno 35,98BTC pryč. Proto jako poslední část analýzy zjistíme, ke kterým adresám byly odeslány a zmapovat tyto trasy.

Cypher:

```
MATCH (a:Address
{AddressID:'1G541ENwQBqG3WZgvYtVCojVgdHFpJ8RXs'}),
(b:Address),

p = allShortestPaths ((a)-[*..4]->(b))

RETURN p;
```

Výsledkem je graf popisující všechny odchozí transakce z dotazem získané Adresy, „1G541ENwQBqG3WZgvYtVCojVgdHFpJ8RXs“.



Obrázek 22 - Graf všech odchozích transakcí adresy

Jak lze vidět na obr. 22 z uzlu vychází celkem 17 plateb do 10 transakcí. Finance z uzlu tedy putovaly dále mezi 12 dalších adres.

8.2 Zhodnocení

Prostřednictvím analýzy jsme názorně předvedli funkčnost porovnávání vzorů (pattern matchingu) v Cypheru za účelem získání pro nás důležitých informací. Díky vhodně zvoleným vzorům, můžeme z databáze získat podstatné informace ať už v oblasti sociálních sítí či např. doporučovacích systémů a dalších oblastí.

Z hlediska rychlosti se na naší databázi prováděli dotazy v intervalech milisekund, složitější dotazy procházející graf do hloubky pak v řádech sekund. Důležitým faktorem je také správné formulování dotazu, používání labelů a specifikování uzlů pomocí příkazu WHERE.

Neo4j je vhodné pro použití v případech, kdy jsou data provázána mnoha vztahy a zkoumání se bude zabývat právě provázaností dat. V dotazech zaměřených na traverzování má Neo4j velmi rychlou odezvu [16] v porovnání například s relačními databázemi avšak v případech použití agregačních funkcí zaměřených na velkou část dat dochází ke zpomalení odezvy.

Velkou výhodou při analýze bylo samotné web rozhraní, ve kterém probíhalo zobrazování výsledků dotazů formou interaktivních grafů nebo tabulek. Toto stále vyvíjející rozhraní je velmi nápomocné jak při práci s daty v databázi tak právě při provádění dotazů.

9 Závěr

V rámci této práce jsme si představili NoSQL databáze, popsali principy, na kterých databáze pracují a důvody jejich vzniku. Po seznámení čtenáře s NoSQL databázemi jsme se zaměřili na jeden z konkrétních typů NoSQL databází, databáze grafové.

Grafové databáze jsou z NoSQL databází schopné pojmout nejmenší objem dat, ale svojí komplexitou přesahují ostatní typy. Díky tomu jsou grafové databáze vhodné pro vysoce provázaná data, která uchovávají ve své přirozené struktuře. Po představení grafových databází a jejich modelu a použití jsme se přesunuli k představení pojmu sociální síť. Sociální sítě obsahují velké množství informací, ale také mají velmi provázanou strukturu dat, proto jsou grafové databáze vhodné pro uchovávání těchto dat. Po představení nejrozšířenějších sociálních sítí a jejich technologií jsme se přesunuli ke konkrétnímu zástupci grafových databází Neo4j.

Neo4j je relativně novou grafovou databází, avšak za dobu své existence si už stihla vybudovat postavení v oblasti grafových databází. Tato databáze ještě stále prochází vývojem a rychle se přizpůsobuje požadavkům moderní doby. Po představení této jistě perspektivní databáze, instalace Serveru a probrání práce s jazykem Cypher jsme se zaměřili na analýzu dat.

Data naší databáze byla získána prostřednictvím serveru Blockchain.info, jde o data bitcoinových transakcí, pomocí kterých byly převáděny transakce z jedné adresy na jinou. V rámci naší analýzy jsme se zaměřili na finance pocházející z šíření ransomwarového viru Cryptowall a názorně jsme předvedli funkčnost použití pattern matchingu pomocí Cypher jazyka.

Tato práce pro mě byla velmi zajímavá s ohledem na to, že jsem si mohl vyzkoušet práci s relativně novou a neustále vyvíjející se technologií, o které si myslím, že se do budoucna stane hojně využívanou technologií v různých oblastech informačních technologií.

Závěrem věřím, že tato práce bude přínosem pro ty, kdo by rádi pronikli do světa Neo4j Serveru a práce s jazykem Cypher.

10 Citovaná literatura

1. **Kumar, Girish.** http://www.3pillarglobal.com/. [Online] 15. 4 2015. http://www.3pillarglobal.com/insights/exploring-the-different-types-of-nosql-databases.

2. —. http://www.3pillarglobal.com/. [Online] 15. 4 2015. http://www.3pillarglobal.com/insights/selection-criteria-for-nosql-database.

3. **Horne, Christopher.** http://www.iqubemarketing.com/. [Online] 12. 4 2015. http://www.iqubemarketing.com/glossary-big-data-terminolgy/.

4. http://cs.wikipedia.org/. [Online] 15. 4 2015. http://cs.wikipedia.org/wiki/Soci%C3%A1ln%C3%AD_s%C3%AD%C5%A5.

5. **Jindra, Martin.** http://www.makevision.net/. [Online] 15. 4 2015. http://www.makevision.net/texty/DP/typologie-socialnich-medii.html.

6. https://managementmania.com. [Online] 15. 4 2015. https://managementmania.com/cs/analyza-socialni-site.

7. http://www.statista.com. [Online] 15. 4 2015. http://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users/.

8. http://en.wikipedia.org. [Online] 18. 4 2015. http://en.wikipedia.org/wiki/Bitcoin_network.

9. http://db-engines.com. [Online] 18. 4 2015. http://db-engines.com/en/ranking_trend/graph+dbms.

10. http://en.wikipedia.org. [Online] 15. 4 2015. http://en.wikipedia.org/wiki/Neo4j.

11. http://neo4j.com. [Online] 15. 4 2015. http://neo4j.com/product/.

12. **Ramba, Jaroslav.** http://www.zdrojak.cz. [Online] 15. 4 2015. http://www.zdrojak.cz/clanky/grafova-terminologie-a-dostupne-technologie/.

13. http://en.wikipedia.org. [Online] 15. 4 2015. http://en.wikipedia.org/wiki/Blockchain.info.

14. **Fauth, David.** https://github.com. [Online] 15. 4 2015. https://github.com/davidfauth/bitcoin-Neo4j.

15. **Wallace, Brian.** http://blog.cylance.com. *Cylance Blog.* [Online] 15. 4 2015. http://blog.cylance.com/tearing-down-cryptowall.

16. **Rodriguez, Marko.** http://java.dzone.com. [Online] 18. 4 2015. http://java.dzone.com/articles/mysql-vs-neo4j-large-scale.

11 Seznam obrázků