1. Úvod
2. NoSQL databáze

2.1 Key-Value úložiště

2.2 Dokumentové orientované databáze

2.3 Grafové database

1. Porovnání s SQL databázemi

3.1 Možnosti a nástroje migrace na NoSQL databázi

1. Praktická část

4.1 Výkonostní testy MongoDB vs. MySQL

4.2 Ukázka použití NoSQL databáze k ukládání binárních dat (souborů)

4.3 Migrace existující MySQL databáze do MongoDB

4.4 Možnosti replikace a škálovatelnosti MongoDB

4.5 Ukládání dat o omezené délce (capped collections - logy apod.)

1. Vyhodnocení výsledků
2. Závěr

1.

obecně o potřebě ukládání dat a databází samotných + big data současné doby

2.

teoretická část - popis hlavních typů NoSQL databází + MongoDB podrobně

3.

MongoDB to MySQL mapping chart, ukázka možností migrace na MongoDB, porovnání vlastností

4.

Testy CRUD operací nad DB (1000 objednávek v e-shopu, rychlá editace, získání velkého množství dat, komplikovaný filtr) CLUSTER?

MongoFiles úložiště uživatelských souborů - jednoduchá implementace, popis chunkování souborů, rychlost přístupu k nim, vyhledávání v nich

Praktická ukázka migrace existující MySQL aplikace do MongoDB

MongoDB sharding, ukázka replikace a běhu v clusteru + CAP teorém

Logování do MongoDB capped collections.

5.

Nejlepší „upgrade cesta na NoSQL“ je do monga, výsledky základních operací MySQL vs. MongoDB (+ clustering?), zhodnocení složitosti implementace jednotlivých serverů. Mongo jako nejvhodnější kandidát pro ukládání struktorovaných dat a to jak systémového charakteru tak jako hlavní DB pro data.

**Katedra informatiky a kvantitativních metod** 

NoSQL Databáze

NoSQL Databases

Autor: Jakub Josef

Vedoucí práce: doc. Ing. Filip Malý, Ph.D.

Akademický rok 2013/2014

**Obsah**

Úvod 1

1. SQL databáze a jejich princip 2

2. NoSQL databáze 3

2.1 Výhody NoSQL databází 4

2.2 Typy NoSQL databází 4

2.2.1 Key-value úložistě 5

2.2.2 Dokumentově orientované databáze 6

2.2.3 Sloupcově orientované databáze 7

3. Porovnání s SQL databázemi 8

4. Praktická část 9

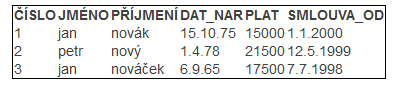
5. Závěr 10

# Úvod

To že počítačové programy vznikají tak, že je určitý tým lidí naprogramuje v nějakém programovacím jazyce, je už prakticky součástí obecného povědomí široké veřejnosti, avšak neméně důležitou součástí téměř každé aplikace, ať už se jedná o webovou stránku nebo klasickou desktopovou aplikaci, je databáze, tedy způsob ukládání nějakých dat, nezbytných pro fungování aplikace jako celku. Dnes v době obrovského rozvoje a rozmachu propojení informačních technologií se životem obyčejného člověka, se značně zvyšují nároky na tyto databáze. Každou sekundou na Internetu přibyde obrovské množství dat, které je třeba někde ukládat a zároveň být schopen v nich velmi rychle vyhledávat. Proto se v posledních letech značně mění náhled na problematiku databázových systémů a můžeme pozorovat odklon od klasických aplikačně složitých SQL serverů k jednodušším řešením. Snížíme tedy složitost aplikace, tak abychom získali větší rychlost a škálovatelnost. SQL je také typickým příkladem relačních databází, kde jednotlivé entity mají definovány relace na jiné, zatímco NoSQL servery nic takového neznají a vzhledem k jejich fungování, ani nepotřebují. Jedním ze stěžejních výhod těchto řešení je absence entitového modelu, tedy každá s entit může mít libovolné množství jakýchkoliv parametrů různých datových typů. Pro tyto aplikačně méně komplexní databázová řešení se vžil název NoSQL databáze a bude se jimi zabývat i moje práce.

# **SQL databáze a jejich princip**

SQL databázi jako typického zástupce relačních databází, kde základním prvkem je tzv. relace kterou můžeme definovat podle Jaromíra Skřivana jako

** „*Relaci, aniž bych zaváděl jakékoliv matematické definice, si lze představit jako tabulku, která se skládá ze sloupců a řádků. Sloupce odpovídají jednotlivým vlastnostem (atributům) entity. Údaje v jednom řádku tabulky zobrazují aktuální stav světa. Budu mít např. tabulku ZAMĚSTNANEC, která bude popisovat entitu pracovníka ve firmě. Sloupce tabulky budou: ČÍSLO, JMÉNO, PŘÍJMENÍ, DAT\_NAR, PLAT a SMLOUVA\_OD. Atribut DAT\_NAR značí datum narození pracovníka a SMLOUVA\_OD uvádí datum, od kterého je pracovník v naší firmě zaměstnán. Představme si, že reálnému světu odpovídá následující naplnění tabulky: “* (1)

Obrázek 1 Databázová tabulka (1)

Data jsou tedy v databázi uspořádána do tabulek, kde každá tabulka má přesně daný počet sloupců, tabulka je tedy modelem uložení dat.

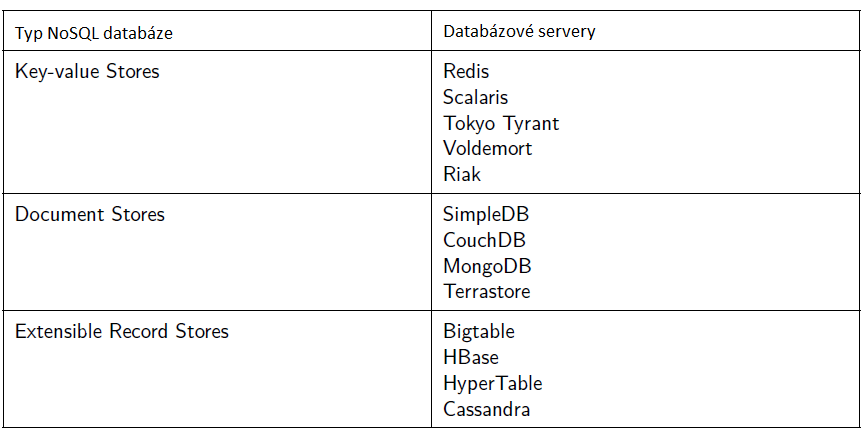
K dotazování (získávání dat) z SQL databáze se používá speciální dotazovací jazyk SQL, který se podle Skřivana skládá z několika částí. (citovat vlastními slovy)

*„První částí jazyka SQL je jazyk* DDL *-* ***Data Definition Language****. Jedná se o jazyk pro vytváření databázových schémat a katalogů. Způsob ukládání tabulek definuje jazyk* SDL *-* ***Storage Definition Language****. Třetí částí pro návrháře a správce je jazyk* VDL *-* ***View Definition Language****, určující vytváření pohledů. Poslední částí, kterou se budu převážně zabývat, je jazyk* DML *-* ***Data Manipulation Language****, který obsahuje základní příkazy INSERT, UPDATE, DELETE a nejpoužívanější příkaz SELECT.“* (1)

# NoSQL databáze

Pojem NoSQL, poprvé použitý Carlem Strozzim v roce 1998 (2), je obecný pojem zahrnující všechny databáze nepostavené na tradičním principu relačních SQL databází popsaného v předchozí kapitole. Zahrnuje tedy velké množství databázových serverů a technologií, které ale vycházejí ze společné myšlenky.

Především je důležité zdůraznit, co NoSQL není – rozhodně to není NO SQL, tedy trendy odmítání relačních databází. Zkratka NoSQL znamená **Not Only SQL**, tedy uvědomění si toho, že relační databáze není jediná možnost řešení persistence, že existují i alternativy, které mohou být v některých případech vhodnější. Zrod mnoha NoSQL databází je spjat s projekty, které se musely vypořádat s obrovským množstvím dat – Facebook (Cassandra), Google (BigTable), Amazon (Dynamo), LinkedIN (Voldemort) atd. (3)

O další rozdělení NoSQL databází se pokusilo už mnoho autorů, jako základní obecné kritérium pro další rozdělení je brán model ukládání dat. Dále rozlišujeme tedy podle Ricka Cattella (4) takto:

Obrázek 2 Rozělení NoSQL databází (4)

## Výhody NoSQL databází

Hlavní výhodou principu NoSQL databází hlavně její jednoduchost a s tím spojená rychlost. Tato hlavní výhoda je ale zároveň i částečnou nevýhodou, jelikož se možnost dotazování velmi omezily, je třeba složitější operace s daty provádět na straně aplikačního serveru vykonávacího obsluhu databáze.

Mezi další výhody patři velmi dobrá podpora horizontálního škálování, tedy běhu na více serverech zároveň. S tím spojená schopnost práce s velmi velkými objemy dat, schopnost jednoduché replikace a flexibilní datové modely (neexistují schémata pro strukturu vstupních dat). Často jsou z těchto důvodů nasazovány na tzv. BigData (poznámka). Řádově větší výkonnost NoSQL řešení se projeví právě u zpracovávání velkých objemů dat. V posledních několika letech se objevilo velké množství NoSQL databázových serverů, které pokrývají celou řadu oblastí při vývoji informačního systému. Můžeme dokonce NoSQL databáze používat jako primární úložiště dat, například MongoDB nabízí vlastní souborový systém (citace?)

## Typy NoSQL databází

NoSQL server je typicky jednoduché úložiště typu klíč -> hodnota, kdy hodnotou může být cokoliv (nejčastěji strukturovaný objektový zápis např. JSON) a klíč bývá unikátní ID vygenerované samotnou databází. Toto je jedna z největších výhod těchto databází, neomezuje totiž programátory žádnými modely předepisujícími strukturu vkládané entity. Typickým zástupcem takto jednoduché databáze je například Redis.

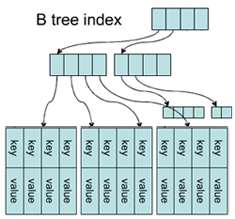
Rozšiřujícím prvkem nad těmito úložišti může být další logické rozdělení uložených hodnot. Toto poskytují například dokumentově orientované databáze (např. MongoDB), kde každá vložená entita je uložena do nějaké kolekce, která popisuje typ této entity. Díky návrhu podobnému SQL (kolekce vs. tabulky), je na tyto databáze nejjednodušší přejít z klasické relační databáze. (podložit citací)

### Key-value úložistě

Key-value úložiště, tedy úložiště kde každá uložená entita je indexována jen a pouze svým unikátním klíčem, nejčastěji nějakým ID. Bývá implementováno jako jednoduchá hash tabulka nebo mapa. (vysvětlit)

Poskytuje jen tři nejzákladnější operace k její obsluze a to:

* Uložení entity pod nějakým klíčem, ten je buď definován ve vstupních datech, nebo vygenerován databázovým serverem
* Získání entity podle klíče
* Smazání entity podle klíče

Tato velká jednoduchost návrhu přináší velmi vysokou rychlost zpracování dat a jednoduchou škálovatelnost, je ale nutné veškeré složitější operace s daty provádět na aplikačním serveru. Databáze tedy skutečně zůstává jen tenkou vrstvou k uložení nějakých dat, která ale díky tomu může běžet na různém počtu strojů podle aktuální potřeby výkonu.

Obrázek 3 Ukázka uložení dat v key-value databázi (7)

Mezi nejznámější key-value úložiště patří Redis, Riak, Voldemort nebo MemcachedDB.

Nejznámější key-value úložiště**:** Riak, Redis, MemcachedDB, Voldemort, Berkeley DB, Hamster DB

### Dokumentově orientované databáze

Dokumentově orientované databáze představují evoluci NoSQL databází, nabízejí kompromis mezi přílišnou jednoduchostí a dostatečnou rychlostí zpracování.

Základním prvkem pro ukládání dat je zde dokument, kterým je míněn objekt s jedinečným identifikátorem a množinou párů klíč-hodnota, jež mohou být do sebe dále libovolně vnořovány. Narozdíl od systémů typu klíč-hodnota většinou umožňují také sekundární indexování atributů, (vysvětlit) a co do funkcionality jsou robustnější. Pro ukládání dat užívají JSON, XML nebo podobné strukturované formáty. (5)

Dalším důležitým prvkem je kolekce, která popisuje a obaluje dokumenty, které jsou v ní uloženy. Vnitřní rozdělení dokumentů do kolekcí doplňuje logické uspořádání entit do celků, podle kterých lze dokumenty zpět získávat. Pro získávání dat už tedy není potřeba žádná složitější aplikační logika, stačí jen získat celou kolekci.

Pro pokročilejší získávání dat lze používat i tzv. matchery, kterými lze definovat kritéria, podle kterých lze výběr omezit. Matchery podobnou funkcionalitou jako SQL příkaz WHERE.Hlavním standardem definujícím komunikaci se serverem je JSON.

*Ukázka databázového dotazu v MongoDB:*

db.bios.find(

{

name: {

first: 'Yukihiro',

last: 'Matsumoto'

}

}

)

*Ukázka odpovědi na ni:*

{

first: 'Yukihiro',

aka: 'Matz',

last: 'Matsumoto'

}

{

last: 'Matsumoto',

first: 'Yukihiro'

}

Nejznámější dokumentově orientované databáze: MongoDB, CouchDB, RavenDB, MarkLogic, TerraStore

### **Sloupcově orientované databáze**

Sloupcově orientované DBMS vycházejí z projektu Google BigTable (6). Jsou opět určeny především vysokozátěžovým systémům a optimalizovány pro paralelní zpracování dotazů. Můžeme si je představit jako klasickou tabulku. Každý záznam (řádek) zde má svůj unikátní identifikátor, ovšem fyzicky se data ukládají nikoliv po řádcích jak je běžné u relačních databází, nýbrž po sloupcích. Je tak rychlejší např. vyhledat záznamy s určitou hodnotou atributu a zároveň se šetří místo na disku, jelikož nemusí být vyhrazeno místo pro prázdné buňky (5)

Nejznámější sloupcově orientované databáze: BigTable, Cassandra, HBase, HyperTable.

# **Porovnání s SQL databázemi**

# **Praktická část**

# **Závěr**

NoSQL databáze neměly a ani nemají složit jako absolutní náhrada klasických relačních SQL databází, to ale nikdy nebyl ani její účel. První implementace vznikaly přímo na míru potřebám projektů, na kterých byly poté nasazeny, přímo na řešení určitých problémů, které vyvstaly. Hlavním důvod jejich vzniku byla potřeba rychle a efektivně pracovat s obrovským množstvím dat. I proto byly průkopníky v této oblasti hlavně velké internetové firmy jako je Google nebo Facebook, které nedokázaly se svými obrovskými datovými objemy rychle a efektivně pracovat pomocí tehdy dostupných technologií. Postupem času se našly další oblasti, kde mohly NoSQL databáze najít své uplatnění. Rychlých key-value úložišť se začalo ve velkém využívat k ukládání velkého množství jednoduchých dat, velmi často jen například id sezení pro každého přihlášeného uživatele. NoSQL databáze našly svoje uplatnění, stejně jako ho našly klasické relační databáze a myslím si, že oba dva přístupy k ukládání a správě dat mohou existovat společně a je velmi těžké rozhodnout, který je ten „lepší“.

**Seznam použité literatury**

1. **Skřivan, Jaromír.** Databáze a jazyk SQL. *Interval.* [Online] 4. Srpen 2000. [Citace: 20. Srpen 2013.] http://interval.cz/clanky/databaze-a-jazyk-sql/.

2. **Strozzi, C.** NoSQL: A Relational Database Management System. [Online] [Citace: 21. Srpen 2013.]

3. **Augustýn, Michal.** Zamyšlení nad NoSQL. *Augiho web.* [Online] 29. Říjen 2011. [Citace: 2013. Srpen 20.] http://www.augi.cz/programovani/zamysleni-nad-nosql/.

4. **Cattell, Rick.** High Performance Scalable Data Stores. [Online] Únor 2010. [Citace: 20. Srpen 2013.] http://cattell.net/datastores/Datastores.pdf.

5. **Regner, Tomáš.** Dokumentově orientované open source databázové systémy. Praha : Vysoká škola ekonomická v Praze, 2012.

6. **Chang, F. et al.** Bigtable: A distrubuted Storage System for Structured Data. [Online] 2006. [Citace: 27. Srpen 2013.]

7. **Tolitius, Anatoly.** Key Value Store List. [Online] 30. Srpen 2009. [Citace: 2013. Srpen 25.] http://www.dotkam.com/2009/08/30/key-value-store-list/.

**Seznam obrázků**

1. Databázová tabulka (1)
2. Rozdělení NoSQL databází (4)
3. Ukázka uložení dat v key-value databázi (7)