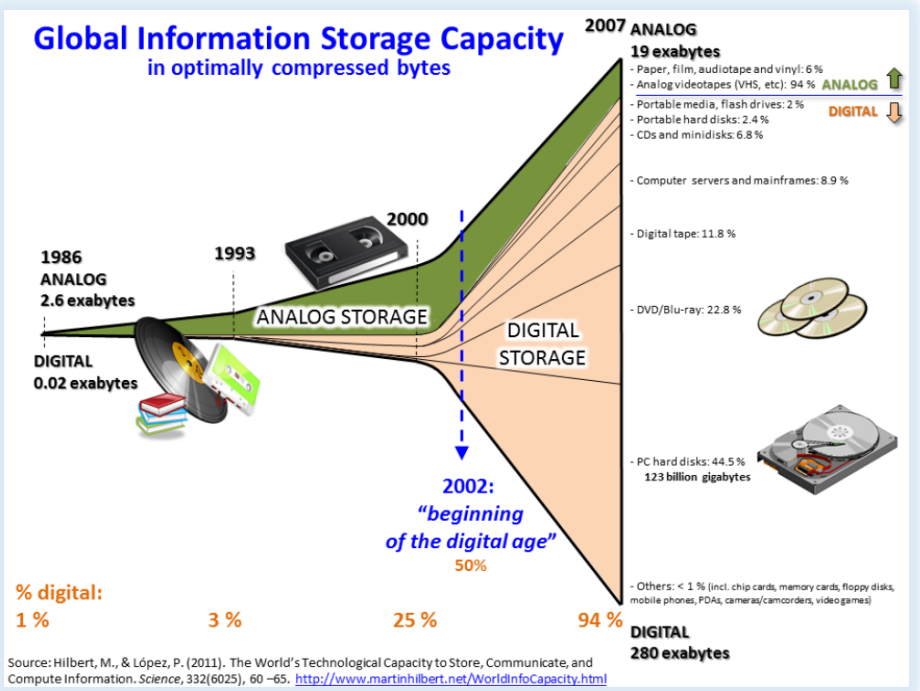
RDB kontra NoSQL



1 Big Data

Historia

- DB
- Data Warehouses Experimenty
- BigData

Hlavné zdroje Big Data

- mobilný Internet
- videa cez Internet
- vyhľadávanie
- sociálne siete

Vlastnosti 3V

- volume
- velocity - generovanie, spracovanie
- variety

Použitie

- dostupnosť
- analýza
- segmentácia – reprezentatívnosť

Technológie

- Ukladanie - distribuované, škálovanie
- Spracovanie - distribuované a paralelné

2 RDB kontra NoSQL

- A.A.Coddov článok z 1970 *A relational model ...*
- Carlo Strozzi v 1998, pojem NoSQL
- Dwight Merriman, Eliot Horowitz, 2007, MongoDB

Viacgeneračný úspech **RDB** (relačných databáz) tkvie v

1) *zabezpečení konzistentnosti dát*
2) *použití sekundárnych kľúčov*
3) *dopytovacom jazyku SQL.*

Tabuľky RDB obsahujú *štrukturované dáta*, zázmamy, riadky, kde stĺpce majú svoj preddefinovaný typ. Zoznam stĺpcov spolu s ich nazvami a názvom tabuľky určuje schému. Relačné databázy párujú dáta z viacerých tabuliek pomocou spoločných atribútov.

NoSQL (Not Only SQL, Non SQL) ukladá **neštruktúrované dáta** bez schémy, zvyčajne **sa vyhýba** operácie **spojenia** a škáluje **horizontálne**.

Ukazuje sa, že RDB systémy sa menej hodia pre

- aplikácie, pracujúce masívnym objemom dát, typy ktorých sa menia rýchlo - semi- a neštruktúrované dáta
- aplikácie, ktoré musia byť neustále zapnuté, prístupnené z mnohých rôznych zariadení a škálované globálne.

Scale up kontra *Scale out*
Namiesto monolitických serverov a ukladacích infraštruktúr sa dnešné aplikácie uprednostňujú **škálovateľné** architektúry podporované **open source** softvérmi.

NoSQL systémy charakterizujú také kľúčové vlastnosti, ako

1) *flexibilný dátový model*
2) *väčšia škálovateľnosť a*
3) *vyšší výkon.*

Všetko má ale svoju cenu. NoSQL systémy zápasia konzistentnosťou alebo menej efektívnym dopytovacím jazykom.

NoSQL systémy môžeme rozlišovať na základe nasledujúcich charakteristík

- Dátový model
- Dopytovací model
- Model konzistentnosti
- API
- Podpora a komunita

- Dátový model

- Dokument modely – MongoDB, Elasticsearch
- Gráf modely – Neo4j, Giraph
- Key-Value modely a modely širokými stĺpcami – Redis, Riak
- Široko-stĺpcové modely – Cassandra, Hadoop/HBase

- Dopytovací model

- Dokument model - celkom bohaté dopytovacie možnosti
- Key-Value modely - rýchle vyhľadávanie pomocou primárneho kľúča, slabé dopytovanie

- Model konzistentnosti

Nerelačné DB systémy zvyčajne manažujú viacnásobné kópie dát kvôli dostupnosti a škálovateľnosti. Preto konzistentnosť dát zaručujú na rôznych úrovniach, nové dáta sa v dopytoch odzrkadľujú s istým oneskorením. MongoDB poskytuje laditeľnú konzistentnosť.

- API

Po dokončení aplikácie, postavenej na konkrétnom type DBS, preniesť ju na iný DB základ je nákladné, náročné a riskantné.

Zrelosť API môže mať značný dopad na čas a náklady potrebné na vývoj a udržiavanie aplikácie a databáz.

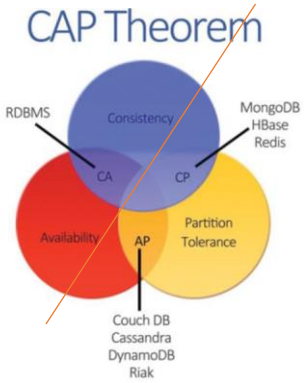
- Podpora a komunita

Databáza so silnou komunitou uľahčuje

- nájsť osvedčené postupy, kódové vzory
- najať vývojára, ktorý ovláda daný produkt.

ACID

- Atomicity - každá transakcia je v celku úspešná alebo vôbec nie
- Consistency - DB z validného stavu sa dostane v dôsledku transakcie do validného stavu (logické požiadavky - constraints, cascades, triggers)
- Isolation - konkurenčné transakcie sa chovajú akoby sekvenčne
- Durability - garancia pre prípad havárie, výpadku prúdu



CAP Theorem (Brewer's Theorem - 2000)

- Consistency – (atomicita a lineárna konzistentnosť)
- Availability
- Partition Tolerance – výpadok časti

Ako zabezpečiť spoľahlivosť systému v prípade straty konzistentnosti?

ACID kontra **BASE** požiadavky

ACID	BASE
Atomicity Consistency Isolation Durable	Basically Available – v zmysle CAP Soft State – zmeny bez vstupu Eventually Consistency – za dĺhší čas

MongoDB poskytuje vysokú dostupnosť, škálovateľnosť a partitioning vďaka dokumentového modelu s dynamickou schémou na úkor konzistentnosti a podpory transakcie.

Dynamická schéma znamená, že dokumenty v kolekcii môžu mať rôzne štruktúry alebo rovnaké kľúče môžu obsahovať dát rôzneho typu.

MongoDB je navrhnutý pre prácu s dokumentmi, ktoré nepotrebujú vopred definované stĺpce alebo typy, vďaka čomu dátový model je veľmi flexibilný.

Podobne ako primárny kľúč RDBMS (ktorý jednoznačne identifikuje každý riadok), MongoDB musí mať kľúč, ktorý jednoznačne identifikuje každý dokument v kolekcii a sa nazýva _id . Jeho hodnota sa buď zadáva manuálne, alebo sa automaticky vygeneruje. Táto hodnota kľúča je nemenná a môže byť akéhokoľvek typu dát s výnimkou polí.

Hodnota pre kľúč sa buď zadáva alebo automaticky generuje jedinečnú hodnotu a priradí sa jej MongoDB. Táto hodnota kľúča je nemenná a môže byť akéhokoľvek typu dát s výnimkou polí.

http://db-engines.com/en/ranking

Rank			DBMS	Database Model	Score	
Apr 2016	Mar 2016	Apr 2015			Apr 2016	Mar 2016
1.	1.	1.	Oracle	Relational DBMS	1467.53	-4.48
2.	2.	2.	MySQL	Relational DBMS	1370.11	+22.39
3.	3.	3.	Microsoft SQL Server	Relational DBMS	1135.05	-1.45
4.	4.	4.	MongoDB	Document store	312.44	+7.11
5.	5.	5.	PostgreSQL	Relational DBMS	303.73	+4.10
6.	6.	6.	DB2	Relational DBMS	184.08	-3.85
7.	7.	7.	Microsoft Access	Relational DBMS	131.97	-3.06
8.	8.	8.	Cassandra	Wide column store	129.67	-0.66
9.	9.	↑ 10.	Redis	Key-value store	111.24	+5.02
10.	10.	↓ 9.	SQLite	Relational DBMS	107.96	+2.19
11.	11.	↑ 14.	Elasticsearch	Search engine	82.58	+2.41

konzistencia - vnútorný súlad
zhodné výsledky - interná konzistentnosť