VŠB TECHNICKÁ UNIVERZITA OSTRAVA
FAKULTA ELEKTROTECHNIKY A INFORMATIKY

Key-value databázové systémy

Key-value database systems

Bc. Jan Jedlička

Bakalářská práce

Vedoucí práce: prof. Ing. Michal Krátký, Ph.D.

Ostrava, 2023

Abstrakt

Cílem diplomové práce je popsat Key-value databázové systémy, ukázat výhodo těchto systémů a představit jedny z jejich významných představitelů. Součástí práce je návrh a implementace testovacího prostředí pro testování těchto systémů s ostatnímy SŘBD. Práce je zakončena vyhodnocením výsledků testů vybraných databázových systémů.

Klíčová slova

NoSQL; Key-value databáze

Abstract

The aim of the diploma thesis is to describe Key-value database systems, to show the advantage of these systems and to present some of their important representatives. Part of the work is the design and implementation of a test environment for testing these systems with other DBMS. The work is finished with an evaluation of the test results of selected database systems.

Keywords

NoSQL; Key-value database

Poděkování

TODO poděkování

Obsah

Seznam použitých zkratek a symbolů

NoSQL	–	No Structured Query Language
Key-value database	–	Klíč-hodnota databáze
TTL	–	Time to live

Seznam tabulek

Kapitola 1

Úvod

NoSQL Key-value (neboli Klíč-hodnota) databáze [1] je jedno z paradigmat pro uložiště dat. Databáze je navržena pro ukládání, načítání a správu různých datových struktur, dnes známých jako slovníky nebo hashovací tabulky. Slovníky obsahují kolekci objektů či záznamů, které mohou opět obsahovat množinu různých polí s daty. Záznamy jsou do slovníků, či hashovacích tabulek, ukládána za pomoci klíče, který identifikuje pozici záznamu v datové struktuře a používá se k následnému vyhledávání dat v databázi.

Key-value databáze fungují velice odlišně než tradiční relační databázové systémy. Relační databáze mají předdefinovanou datovou strukturu v databázi jako sérii tabulek s dopředu definovanými datovými typy. Díky tomuto modelu může relační databázový systém provádět řadu optimalizací. Na druhou stranu Key-value databázové systémy mohou mít pro každý záznam různě definované kolekce dat s odlišnými velikostmi a počty atributů. Tato vlastnost nabízí Key-value databázovým systémům flexibilitu a možnost přiblížení se k objektově orientovanému programování. Protože Key-value databáze nevyžaduje pevně nastavené datkové typy hodnot, jako je tomu u relační databáze, tak Key-value databáze často potřebují méně paměti k uložení stajných dat, což může vést k značnému nárůstu výkonu. Dále tyto databáze bývají distribuované a dosahují horizontální až lineární škálovatelnosti.

Výkon a nedostatečná standardizace omezovaly Key-value databázové systémy pouze na specializovaná využití, ale díky rychlému přechodu na cloud computing dochází v posledních letech k rozšíření obecné využitelnosti NoSQL databázových systémů. Například databázový systém Redis [2] je v současnosti jedním z deseti nejlépe hodnocenách [3] databázových systémů napříč relačními i NoSQL databázovými systémy.

Kapitola 2

NoSQL Key-value databázové systémy

V současné době existuje nespočet různých Key-value databázových systémů, od malých open source projektů po velké placené cloud služby. Různé systém disponují odlišnými vlastnostmy jako je propustnost, škálovatelnost, uživatlká přívětivost skrz dotazovací jazyk a podporu aj. Dle průzkumu [4, 5, 3] bylo vybráno 8 aktuálně významných Key-value databázových systémů se snahou o jednoduchý popis, porovnání (2.1) a konešný výběr vhodnách Key-value databázových systémů se snahou otestovat vlastnosti těchto systémů.

2.1 Amazon DynamoDB

Amazon DynamoDB [6] je v současné době největší a nejvyužívanější Key-value databázový systém. Jedná se o serverless cloud systém s odezvou v řádu jednotek mikrosekund a využitím v oblastech jako je web tech, IoT, mobile a gaming. DynamoDB je plně a automaticky spravovatelná, multi master databáze zaměřené na vysoké využití horizontální škálovatelnosti. Unikátní primární klíče umožňují identifikaci jednotlivých záznamů v tabulkách a sekundární index zlepšuje dodazovací flexibilitu. Primární klíč je jako vstup do hashovací funkce a výsledý hash nám udává fyzickou pozici uloženého záznamu. DynamoDB poskytuje silnou konzistenci na čtení hodnot od posledních aktualizací. Atomické čítače umožňuí automatické změny hodnot číselných atribnutů. Využívá TTL pro prošlé záznamy v tabulkách. Archivace dat je umožněna díky full backupu. Amazon DynamoDB má i VPC pro soukromou komunikaci bez potřeby využití internetu.

Databázový systém má konzolové API pro správu databáze a práci s daty, systém však nabízí i možnost využití jazyku PartiQL [7] který je vhodný pro kompatibilní SQL dotazy na schemaless databázích. DynamoDB API je rozděleno do čtyř hlavních částí. Kontrolní plán, který zahrnuje funkce spojené s vytvářením, upravováním, mazáním a získáním jmen všech tabulek. Dále umožňuje výpis podrobných specifikací dané tabulky, jako jsou primární klíče, indexy a nastavení propustosti. Následuje datový plán, který poskytuje CRUD operace pro data v dané tabulce. S daty je možno pracovat jednotlivě po záznamech a nebo pomocí Batch funkcí, které nám dovolují provést stejnou

operaci nad desítkami záznamů v jednu chvíli a dosahují tak vyšší propustnosti než při volání stejných funkcí pro jednotlivé záznamy opakovaně. Následně je možná provést Scan pro získání všech záznamů dané tabulky nebo indexu případně Query pro obdržení hledané části dat. Třetí část je DynamoDB Streams pro práci s časovými sequencemi a práci s logy za posledních 24 hodin. Stream API poskytuje funkce pro výpis všech streamů, konkrétní popis daného streamu, získání iterátoru pro daný stream a nakonec získání jednoho záznamu z daného streamu. A poslední částí API jsou ACID transakce. Transakce jsou rozděleny do dvou částí, první část je určena pro batch vkládání, úpravu a mazání záznamů a druhá část slouží k batch získání záznamů.

2.2 Oracle NoSQL Database

Oracle NoSQL Database [8] je databázová cloud služba vhodná pro práci s velkými objemy dat a odhadovatelnou nízkou odezvou v řádu jednotek milisekund. Služba je postavena na enginu z Oracle Berkeley DB. Databáze je plně spravovatelná, flexibilní, škáluje horizontálně, dynamicky a dosahuje vysokých výkonů. Mimo Key-value data se jedná i o spolehlivé uložiště pro dokumenty a data s pevně daným schématem. Vzhledem k tomu že databázový systém je plně spravovaný společností Oracle, tak je pro vývojáře rychlé a snadné začít tuto službu využívat a soustředit se pouze na vývoj aplikací, neboť není potřeba se obtěžovat se správou základní infrastruktury databáze, softwaru, zabezpečení atp. Jedná se o Single Master, Multi Replica grafový systém. Pokud dojde k chybě na masteru, je master automaticky nahrazen jednou z replik. Pro Key-value ukládání s kapacitu jednotek terabytů využívá systém velký počet Storage uzlů, které je možno skupinově konfigurovat. Pro udržení konzistence jsou Storage uzly replikovány. Uzly a hrany v grafu reprezentují entity které vytvářejí vztahy a propojení. Sdílený systém, uniformně alokuje data okolo ostatních částí skupin. Databáze obsahuje i SQL Query s jazykem pro import, export a přenos dat mezi různými Oracle NoSQL databázemi. Mimo jiné je zde podpora i pro Failover, SwitchOver, Bulk Get API, Off Heap Cache a podpora Big Data SQL.

Restové API pro Oracle NoSQL Database je rozděleno do pěti částí. Správa indexů, která dovoluje vytvářet a mazat indexy pro danou tabulku. Tato část API také umožňuje zobrazit všechny indexy které jsou pro danou tabulku vytvořeny a společně s detailním popisem každého indexu. Druhá část API se věnuje dotazům, umožňuje tedy syntaktickou kontrolu daného SQL dotazu, předpřipravení a spuštění dotazu. Třetí část je zaměřena na správu záznamů, obsahuje tedy CRUD funkce pro jednotlivé záznamy. Tato část ale neobsahuje funkci pro úpravu existujícího záznamu a ani neumožňuje správu mnoha záznamů najednou, pro úpravu je tedy nutno provést funkci odstranění záznamu a vložení nového a všechny záznamy je tedy také potřeba spravovat jednotlivě a postupně. Čtvrtá část je zaměřena správě tabulek, obsahuje možnost vytvoření, upravování, a mazání tabulek. Tato část také umožňuje výpis všech tabulek, informace o dané tabulce a využívání dané tabulky. Poslední část API se věnuje správě pracovních požadavků, lze zde zobrazit

stav jednotlivých požadavků, mazat požadavky, získat chyby či log daného požadavku a list všech požadavků.

2.3 Redis

Redis [2] je in-memory uložiště pro datové struktury, využívané jako Key-value databáze, cache, streaming engine nebo zprostředkovávatel zpráv. Toto datové uložiště má skvělé využití pro klíče v podobě hashe a hodnoty jako velký json objekt. Pro perzistenci dat můžeme ukládání dat na disk provádět po nasatvitelných pravidelných intervalech, nebo je možné data logovat vždy při vykonávání operací, pokud nemáme zájem o trvanlivost dat je možné ukládání dat vypnout úplně a datové uložioště využít čistě jako cache. Uložiště škáluje horizontálně. Redis podpuruje datové struktury jako stringy, hashe, listy, množiny, bitmapy, hyperloglog a geospatial indexy. Nad datovými typy Redis umožňuje rychlé atomické operace jako rozšíření stringů, přídání prvků na začátek a konec listů atd. Datové uložiště také poskytuje setřízené množiny pro vytváření indexů dle ID nebo jiného číselného atributu. Redis hashing ukládá data jako klíč a mapu. Keyspace notifikace dovoluje klientům odebírat Publisher-Subscriber kanály. Pro práci s dotazy na souřadnice a geometrii je možné využívat Geo API. Redis umožňuje provádět transakce, volat lua skripty a nastavovat různé úrovně TTL pro záznamy. Redis podporuje Trivial-to-setup Master-Slave asynchronního replikování, společně s rychlou neblokující se prvotní synchronizaci. Struktura pro ukládání dat je Single-rooted replikovaný strom. Redis má vlastní API pro práci s daty pro populární programovací jazyky jako C, Python, Java a Javascript.

S Redis databází lze pracovat například pomocí konzolového rozhraní, toto CLI [9] poskytuje řadu jednoduše čitelných ale netradičních příkazů pro práci s daty. Vždy potřebujeme specifikovat klíč s jehož daty chceme v databázi pracovat. Pomocí příkazu SET a DEL vkládáme do databáze či mažeme jednotlivé hodnoty pro zvolený klíč. Příkazem GET získáme hodnoty pro daný klíč, případně můžeme zjistit zda již existuje záznam pro daný klíč příkazem EXISTS. Pokud vyžadujeme práci s poli tak můžeme pro daný klíč zleva i zprava vkládat hodnoty zřetězené v poli díky příkazům LPUSH a RPUSH. Obdobně odebíráme hodnoty z pole pomocí LPOP a RPOP, příkazem LRANGE vypíšeme hodnoty z pole a příkazem LLEN zjistíme počet jeho záznamů. Místo jednoduchých polí je možno pracovat i s množinami pomocí příkazů SADD, SREM, SISMEMBER a obdobně, množiny mohou být i setřízené a pro ně se využívají příkazy jako ZADD. Pro práci se záznamy strukturovaných jako kolekce párů atribut-hodnota se využívá datový typ Hash, umožňuje nám pro daný klíč uložit záznam obsahující názvy atributů a jednotlivé hodnoty pro ně. Opět se zde využívají příkazy jako HSET a HGETALL pro nastavení a získání daného záznamu, případně HGET pro získání hodnoty daného atributu pro záznam na zadaném klíči. API obsahuje také příkazy pro ostatní datové typy jako jsou Bitmapy, Geoprostory, HyperLogLog a další.

2.4 Aerospike

Aerospike [10] je Key-value databáze využívající Hybrid Memory architekturu [11], která umožňuje odezvu do jednotek milisekund a vysokou propustnost v řádu stovek tisíc až milionu operací za sekundu. Hybrid Memory architektura od Aeorispike je implementována tak, že index je čistě In-Memory, tím pádem není index persistentní (vhodné například pro uživatelské cache sessions), a data jsou uložena čistě persistentně na SSD disku a čtou se přímo z něj. Díky tomu že je Aerospike jako Key-value databáze naprosto Schena-less tak je možné definovat Sets a Bins za běhu pro maximální flexibilitu aplikací. Databáze škáluje lineárně a poskytuje silnou konzistenci, nízkou cenu a korektnost. Umožňuje real-time analýzu pro rychlé rozhodování a dynamickou optimalizaci pro vhodné využívání zdrojů dat, proto je databáze vhodná pro velké a stále aktualizované databáze. Poskytuje server-side clustering, bezpečnost na transportní vrstvě. Databáze také umožňuje Customer deployment s nulovým downtime. V praxi se Aerospike díky svým vlastnostem využívá například pro banking, telekomunkace, adtech a gaming. Aerospike poskytuje vlastní silný dotazovací jazyk AQL [12], který má prakticky shodnou syntaxi jako SQL (i když se o SQL nejedná). Vlastní vytvořitelné agregační funkce pomocí Lua jazyku (flexibilní pro agregační algoritmy).

Dotazovací jazyk AQL se snaží zachovat standartní SQL syntaxi, obvyklé příkazy SELECT, INSERT, DELETE jsou tedy zachovány. Je možné i vytvářet vlastní indexy nad tabulkami pomocí CREATE INDEX a provádět agregace pomocí AGGREGATE. Pro dotazy nad konkrétním záznamem specifikovaným pomocí hexadecimálního řetězce či Base64 lze v podmínce dotazu použít porovnání hodnoty s DIGEST nebo EDIGEST. Dotazování můžeme provádět i standartně nad primárním klíčem a ostatními atributy. Při vkládání záznamů lze specifikovat speciální datové typy atributů jako je LIST, MAP, GEOJSON a další.

2.5 Oracle Berkeley DB

Oracle Berkeley DB [13] je rodina vestavěných Key-value databázových knihoven. Jedná se o čistě In-memory databázi, díky čemuž dosahuje vysokého výkonu a odezvy v jednotkách mikrosekund. Databáze škáluje horizontálně. Data jsou replikována pro vysokou dostupnost z vícero zdrojů a dobrou toleranci chybovosti. Oracle Berkeley DB využívá vhodné datové struktury pro práci s daty jako je B-strom, hash table indexy nebo fronta. Databáze využívá obnovitelné ACID transakce a poskytuje několik různých úrovní izolace (včetně MVCC [14]). Data jsou dělena do oddílů dle key ranges. Umožňuje komprimaci dat. Databáze je Single-master, Multi-replica, tedy je vysoce dostupná a umožňuje dobrou konfigurovatelnost. Repliky umožňují čtecí škálovatelnost, rychlý fail-over, hot-standby a další distribuované konfigurace dodávající podnikové prostředky v malém, vestavěném balíčku. Pro přístup k datům a nastavení databáze se využívá jednoduché volání funcion-call API. Spousta moderních programovacích jazyků, jako například C++, C#, Javy, Python atp. podporuje tyto knihovny. Data mohou být ukládána v nativním formátu aplikace, XML, SQL nebo jako

Java Objekty. Oracle Berkeley DB je vhodný nástroj pro vše od lokálního uložiště po world-wide distribuovanou databáze (od kilobytů po petabyty).

Interakce s Berkley DB SQL API je prakticky identická jako s SQLite [15]. Pro práci s databází vytvořenou rozhraním BDB SQL používáte stejná rozhraní API, stejné Shell prostředí, stejné příkazy SQL a stejné PRAGMA, jako se využívá u SQLite. BDB SQL rozšiřuje standartní SQLite PRAGMA o možnosti nastavení velikosti alokované paměti sdílených zdrojů, nastavení počtu bucketů v hashovací tabulce objektů zámků, zvolení soukromého prostředí místo sdíleného, přesměrování logování chyb do vlastního souboru, nastavení příznaku, který způsobí, že sdílené prostředky databáze budou vytvořeny ve sdílené paměti systému a další. Dalším drobným rozdílem je že BDB SQL rozhraní nepodporuje klíčové slovo IMMEDIATE.

2.6 Riak KV

Riak KV [16] je distribuovaná NoSQL Key-value databáze s pokročilou lokání a multi-cluster replikací, která garantuje čtení a zápis i v případě selhání hardwaru nebo síťových oddílů. Riak využívá bezkonfliktní replikované datové typy (CRDT [17]), které umožňují nezávisle a souběžně aktualizovat jakoukoliv repliku v distribuované databázi se zajištěním sjednocení hodnot pomocí algoritmu který je součásti samotného datového typu (flagy, registry, čítače, množiny a mapy). Poskytuje konfiguraci aktivního clusteru a dosahuje nízké latence v řádu jednotek milisekund díky dodávání dat z nejbližšího data centra. Databáze rozděluje data z clusterů pro své dostupné zóny, má multi claster repliky a využívá redundance dat v geografickém regionu. Riak tedy automaticky distribuuje data skrz cluster, pro robustnost a vysoký výkon. Key-value databáze poskytuje flexibilní datový model bez předem definovaného schématu. Databáze vylepšené logování chyb a reporty. Data jsou automaticky komprimována pomocí Snappy kompresní knihovny [18]. Databáze využívá master-less architekturu, je vysoce dostupná a má design horizontální škálovatelnosti. Škálovatelnost je téměř lineární při využití snadného přidání hardwarové kapacity bez nutnosti mnoha operací. Riak KV dovoluje zpracování dat pro analýzu a vyvození závěrů pro zlepšení chodu databáze. Riak KV je navržen pro nulové restrikce na hodnoty, takže session data mohou být enkódována mnoha způsoby a nevyžadují změnu schématu. Během nejvyšší zátěže nezhoršuje databáze zápis a horizontální škálovatelnost, uživatelé jsou stále obsluhování bez problémů. Databáze je vhodná pro ukládání velkého množství nestrukturovaných dat, také pro big-data aplikace, ukládání dat z připojených zařízení a replikování dat do okolí. Díky nízké latency je databáze vhodná i pro chat/messaging aplikace. Riak KV exceluje v soukromém, veřejném či hybridním cloud nasazování.

Riak KV API obsahuje všechny potřebné CRUD operace pro správu objektů. Při vytváření nových objektů je potřeba nastavit typ a název bucketu, který skladuje klíče a data do něj vložená, bucket má také vlastní indexy pro vyhledávání dat uvnitř něj. Dva různé buckety mohou uchovávat stejnou hodnotu klíče, ale jeden bucket obsahuje pouze unikátní klíče. Klíč pro data lze specifikovat explicitně vlastní při vytváření objektu pomocí parametru nebo při jeho absenci je datům přiřa-

zen klíč náhodný. Při vkládání dat do databáze můžeme jednoduše nastavit parametr TTL daného objektu a také počet jeho replik. Při čtení dat můžeme před získáním výsledku zadat minimální počet replik, které se musí shodnout na stejných datech pro zvolený klíč. Pro efektivnější dotazy lze vytvořit vlastní indexy pro výchozí nebo námi zvolené datové schéma. Lze se dotazovat na data pro zvolený klíč nebo provádět fulltextové vyhledávání. Databáze poskytuje i funkce pro tvorbu sekundárních indexů a následné dotazy nad nimi. Riak API také umožňuje hlubší nastavení autorizace a bezpečnosti, práce s replikami a řešení konfliktů.

2.7 Voldemort

Project Voldemort [19] je distribuovaná Key-value databáze založena na Amazon DynamoDB. Škáluje horizontálně pro čtení i zápisu. Umožňuje zapojení storage-enginu (MySQL, Read-Only). Databáze automaticky replikuje data napříč servery pro dostupnost a bezpečnost jednotlivých oddílů při vysoké propustnosi, nicméně každý server obsahuje pouze část z celkových dat. Databáze je decentralizovaná z pohledu uzlů, každý uzel je samostatný a nezávislý, nenáchází se zde žádný centrální řídící uzel nebo uzel řídící řešení chyb. Voldemort má výkonost desítek tisíc operací za sekundu na jeden uzel (1 op. za 50 mikro sekund), samozřejmě závisí na hardwaru, síti, systému disku atp. Konzistence dat je nastavitelná (přísné kvórum nebo případná konzistence). Selhnání serverů jsou ošetřována transparentně, pro lepší viditelnost, interní monitorování a validaci dat lze využívat JMX [20]. Data jsou verzována pro maximální integritu i během poruch. In-Memory caching pro eliminaci oddělených částí cache, jednoduché a rychlé in-memory testování (např. pro unit testy). Databáze umožňuje jednoduchou distribuci dat skrz stroje, data mohou být rozdělována například dle primárních klíčů. Databáze má hashovatelné schéma, vyhledávání dle primárního klíče a možnost modifikace jednotlivých hodnot. Voldemort poskytuje široké možnosti pro klíče i hodnoty díky serializaci včetně listů a tuplů s pojmenovanými poli. Pro serializaci (Java Serialization, Thrift, Avro) se využívá JSON datový model v kompaktním bytovém formátu, probíhá zde typová kontrola dat dle očekávaného schématu. Pomocí API je možné rozhodovat o replikování a místech ukládání dat, nastavení různé strategie pro specifické aplikace a možnost distribuce dat skrz data centra která jsou mezi sebou geologicky velice vzdálená.

2.8 InfinityDB

InfinityDB [21] je NoSQL hierarchicky tříděná Key-value databáze implementovaná v Javě. Databáze má možnost využít čistě In-Memory ukládání dat která je vhodné pro cache, nebo naopak se mohou data ukládat i perzistentně na disk do souboru, nastavení měnit bez zasahování do kódu. Přístup k datům v cachy je plně více vláknový, vužívá se většina jader, a data která nejsou často využívaná jsou stránkována na disk. Výkon v jednotkách miliónů operací za sekundu pro více vláknové operace v cachy. Veškerá data a informace o databázi jsou na disku uložena v jednom souboru, jsou

proto stále aktuální, čímž je docílena maximální bezpečnost, korektnost. Databáze je designovaná právě pro použití jen jednoho kompletního souboru s okamžitým zotavením a nevyžaduje proto administraci. Databáze neobsahuje dodatečné konfigurační nebo dočasné soubory, upgrade skripty a ani logy. Zotavení je tady bez logů o transakcích okamžité ihned po restartu. V databázi není potřeba dělat čištění junk souborů po operacích když zde nejsou žádné zanechány. InfinityDB podporuje ACID pro vlákna a ACD pro bulk operace. Databáze poskytuje prostor pro ukládání strukturovaných, polostrukturovaných a nestrukturovaných dat, tento jednoduchý model umožňuje ukládání vnořených Multi-values, je možné reprezentovat stromy, grafy, Key/Value mapy, dokumenty, velká řidká pole a tabulky. Schema je možné měnit za běhu pro zpětnou i následující kompatibilitu. Data dotazů lze dynamicky sledovat (set logic views, delta views, ranges). InfinityDB poskytuje jednoduché API o deseti základních příkazech (insert, delete, delete suffixes, update, first, next, last, previous, commit, rollback). Databáze má využití pro servery, pracovní stanice a příruční zařízení.

2.9 Nezmíněné významné KV DB v 2022

- MongoDB (? Document DB)
- Microsoft Azure Cosmos DB
- Couchbase (Key-value jako sekundární model, primárně Document DB)
- Cassandra (Wide column store)

Tabulka 2.1: Porovnání Key-value databází

Databáze	Amazon DynamoDB	Oracle NoSQL DB	Redis	Aerospike	Oracle Berkeley DB	Riak KV	Voldemort	InfinityDB
Čistě cloud	ano	ne	ne	ne	ne	ne	ne	ne
Schéma dat	ne	ano i ne	ne	ne	ne	ne	ne	ano
Licence	komerční	open source	open source	open source	open source	open source	open source	komerční
Server OS	hostovaná	Linux, Solaris	Linux, Windows, OS X, BSD	Linux	Linux, Windows, OS X, Android ad.	Linux, OS X	Linux, Windows	Linux, Windows, OS X, Solaris
Napsáno v	-	Java	C	C	C, C++, Java	Erlang	Java	Java
Sekundární indexy	ano	ano	ano	ano	ano	omezené	ne	ne
Koncept transakcí	ACID	ACID v rámci uzlu	atomické, izolované	atomické	ACID	ne	ne	ACID
Triggery	ano	ne	pub/sub	ne	ano	ano	ne	ne
Dělící metody	sdílení	sdílení	sdílení	sdílení	ne	sdílení	ne	ne
Replikační metody	ano	source-replica multi-region	source-replica, multi-souce	volitelná faktor repl.	source-replica	volitelný faktor repl.	ne	ne
Administrace	vysoká	nízká	vysoká	vysoká	vysoká	vysoká	vysoká	ne
Škálovatelnost	horizontální	horizontální	horizontální	lineární	horizontální	lineární	horizontální	horizontální
Odezva	mikrosekundy	milisekundy	milisekundy	milisekundy	mikrosekundy	milisekundy	milisekundy	milisekundy
Dotazovací jazyk	PartiQL	Omezený SQL	Redis query	AQL	SQL	Riak query	Voldemort query	InfinityDB query

Kapitola 3

Prostředí pro testování databázových systémů

TODO

Kapitola 4

Vyhodnocení výsledků testů

TODO

Kapitola 5

Závěr

TODO

Literatura

1. *Key–value database* [online]. 2022. [cit. 2022-11-18]. Dostupné z: https://en.wikipedia.org/wiki/Key%E2%80%93value_database.
2. *Redis* [online]. 2022. [cit. 2022-11-18]. Dostupné z: https://redis.io/.
3. *DB-Engines Ranking* [online]. 2022. [cit. 2022-11-18]. Dostupné z: https://db-engines.com/en/ranking.
4. *Top NoSQL Key Value store Databases: Predictiveanalyticstoday* [online]. 2022. [cit. 2022-11-13]. Dostupné z: https://www.predictiveanalyticstoday.com/top-sql-key-value-store-databases/.
5. *Best Document Databases: G2* [online]. 2022. [cit. 2022-11-13]. Dostupné z: https://www.g2.com/categories/document-databases.
6. *Amazon DynamoDB* [online]. 2022. [cit. 2022-11-18]. Dostupné z: https://aws.amazon.com/dynamodb/.
7. *PartiQL* [online]. 2023. [cit. 2023-01-09]. Dostupné z: https://partiql.org/docs.html.
8. *Oracle NoSQL Database* [online]. 2022. [cit. 2022-11-19]. Dostupné z: https://www.oracle.com/database/nosql/technologies/nosql/.
9. *Redis CLI* [online]. 2023. [cit. 2023-01-14]. Dostupné z: https://redis.io/docs/manual/cli/.
10. *Aerospike* [online]. 2022. [cit. 2022-11-20]. Dostupné z: https://aerospike.com/.
11. *Hybrid Memory Architecture* [online]. 2022. [cit. 2022-11-20]. Dostupné z: https://aerospike.com/products/features/hybrid-memory-architecture/.
12. *Aerospike Quick Look (AQL)* [online]. 2022. [cit. 2022-11-20]. Dostupné z: https://docs.aerospike.com/tools/aql.
13. *Oracle Berkeley DB* [online]. 2022. [cit. 2022-11-21]. Dostupné z: https://www.oracle.com/database/technologies/related/berkeleydb.html.
14. *Multiversion concurrency control* [online]. 2022. [cit. 2022-11-21]. Dostupné z: https://en.wikipedia.org/wiki/Multiversion_concurrency_control.

15. *SQLite* [online]. 2023. [cit. 2023-01-15]. Dostupné z: https://www.sqlite.org/index.html.

16. *Riak KV* [online]. 2022. [cit. 2022-11-21]. Dostupné z: https://riak.com/products/riak-kv/index.html.

17. *Conflict-free replicated data type* [online]. 2022. [cit. 2022-11-21]. Dostupné z: https://en.wikipedia.org/wiki/Conflict-free_replicated_data_type.

18. *Snappy* [online]. 2022. [cit. 2022-11-21]. Dostupné z: https://www.solvusoft.com/cs/file-extensions/software/google/snappy/.

19. *Project Voldemort* [online]. 2022. [cit. 2022-11-22]. Dostupné z: https://www.project-voldemort.com/voldemort/.

20. *Java Management Extensions* [online]. 2022. [cit. 2022-11-22]. Dostupné z: https://www.oracle.com/technical-resources/articles/javase/jmx.html.

21. *InfinityDB Java NoSQL Database: Boiler Bay Software* [online]. 2022. [cit. 2022-11-22]. Dostupné z: https://boilerbay.com/.

22. *How do NoSQL databases work? Simply Explained! youtube* [online]. 2021. [cit. 2022-11-18]. Dostupné z: https://www.youtube.com/watch?v=0buKQHokLK8.