UNIVERZITA HRADEC KRÁLOVÉ  
Fakulta informatiky a managementu

*Nasazení a inicializace MongoDB databáze*

*Bc. Martin Malíř, 1. ročník*

V Hradci Králové dne 14. 05. 2025

Obsah

[Úvod 3](#_Toc192620999)

[1 Architektura 4](#_Toc192621000)

[1.1 Schéma a popis architektury 4](#_Toc192621001)

[1.2 Specifika konfigurace 5](#_Toc192621002)

[1.2.1 CAP teorém 5](#_Toc192621003)

[1.2.2 Cluster 5](#_Toc192621004)

[1.2.3 Uzly 5](#_Toc192621005)

[1.2.4 Sharding 5](#_Toc192621006)

[1.2.5 Replikace 5](#_Toc192621007)

[1.2.6 Perzistence dat 6](#_Toc192621008)

[1.2.7 Distribuce dat 6](#_Toc192621009)

[1.2.8 Zabezpečení 6](#_Toc192621010)

[2 Funkční řešení 7](#_Toc192621011)

[2.1 Struktura 7](#_Toc192621012)

[2.2 Instalace 7](#_Toc192621013)

[3 Případy užití a případové studie 8](#_Toc192621014)

[4 Výhody a nevýhody 9](#_Toc192621015)

[5 Další specifika 10](#_Toc192621016)

[6 Data 11](#_Toc192621017)

[7 Dotazy 12](#_Toc192621018)

[Závěr 13](#_Toc192621019)

[Zdroje 14](#_Toc192621020)

[Přílohy 15](#_Toc192621021)

Úvod

Tato práce se zabývá vytvořením a inicializací databáze MongoDB, která je realizována jako sharded cluster. V práci je popsáno řešení od základního teoretického návrhu, až po detailní konfiguraci a zabezpečení.

TODO: rozepsat více

# Architektura

Tato kapitola popisuje nasazení MongoDB databáze v rámci sharded clusteru pomocí Docker Compose.

## Schéma a popis architektury

Obsah obrázku snímek obrazovky, text, software, design

Obsah vygenerovaný umělou inteligencí může být nesprávný.

MongoDB cluster v této architektuře obsahuje jeden config server, který ukládá metadata o struktuře clusteru a rozdělení dat. Tento server se pojí na replica set o třech nodách, kde jedna vystupuje v roli primárního uzlu s dvěma sekundárními uzly. Tyto uzly obsahují samotná data, která jsou rozdělena mezi jednotlivé shardy pro horizontální škálování. V poslední řade, je zde router pro obsloužení komunikace mezi klientem a zbytkem databáze. Architektura byla navržena pro vysokou dostupnost, škálovatelnost a bezpečnost s využitím keyFile autentizace.

Od doporučeného použití se implementace liší implementací tří samostatných shardů v rámci jednoho Docker Compose souboru, což usnadňuje vývoj i testování, avšak není doporučeno kvůli produkčním i provozním omezením. Například, v této konfiguraci, běží všechny komponenty na jednom serveru, který při selhání kompletně znemožní používání databáze.

## Specifika konfigurace

Xxxxxx

### CAP teorém

MongoDB je ve své základní konfiguraci zaměřeno na konzistenci (**C**AP – consistency). Pokud je proved zápis do databáze a následné čtení, tak za předpokladu úspěšného zápisu, vždy dostaneme na výstupu očekávaná data. To je splněno díky tomu, že MongoDB vystupuje v takzvaném režimu „single-master“, ve kterém jsou všechny operace provedené pomocí primárního uzlu přeneseny na sekundární uzly. To v praxi znamená, že pokud primární uzel selže, data nebudou propagována dál a nebude zvolen jiný primární uzel, dokud není zaručené doručení a konzistence dat.

Nicméně, MongoDB je možné nakonfigurovat tak, aby byla zajištěna dostupnost (C**A**P – availability) i odolnost proti dělení (CA**P** – partition tolerance). V tomto případě, je povoleno čtení i ze sekundárních uzlů, a to i v případě, že dojde k de-synchronizaci s primárním uzlem. Tímto způsobem lze zajistit dostupnost dat, i v případě, že primární uzel selže, avšak je potřeba mít na paměti implicitně způsobené problémy ztráty synchronizace – kde data nemusí být napříč všemi uzly stejná.

Tato konfigurace clusteru zajišťuje garanci konzistence a dostupnosti. Odolnost proti dělení je implicitně podporována pomocí replikace a shardingu. V případě selhání primárního uzlu budou data doručena pomocí jednoho ze sekundárních uzlů. Díky tomu je poskytnut konzistentní pohled na data i při výpadku některého z uzlů.

### Cluster

Cluster je skupina propojených vzájemně kooperujících databázových (nebo jiných) serverů. Tyto servery se vzájemně podílejí na zpracování a ukládání dat. Cílem clusteru je zajištění škálovatelnosti, vysoké dostupnosti a odolnosti vůči výpadkům. MongoDB používá cluster ke správě velkých objemů dat a jejich efektivnímu rozdělení mezi jednotlivé uzly. Každý cluster by se měl skládat z komponent:

* Config server
* Router (mongos)
* Shardy

Tyto komponenty společně tvoří distribuovaný systém.

V tomto řešení je použit jeden cluster, který nabízí centralizovanou správu a řízení distribuce dat. Jmenovitě pak:

* Zjednodušuje konfiguraci a administraci clusteru
* Umožňuje snadné řízení replikace a shardingu
* Snižuje režijní náklady

### Uzly

Každý cluster MongoDB se skládá z několika uzlů. V minimální konfiguraci je to:

* Config server
* Router (mongos)
* Shardy

Config server udržuje metadata o clusteru a o rozdělení dat mezi jednotlivé shardy. Router slouží ke směrování požadavků od klienta na správné shardy. Shardy jsou hlavní datové uzly, kde jsou fyzicky uložena data. Každý shard je replikovaný set zajišťující redundanci dat.

Dostatečný počet uzlů je důležitý pro zajištění škálovatelnosti a dostupnosti systému. Větší počet uzlů umožňuje distribuované zpracování požadavků, což zlepšuje výkon a zajišťuje odolnost proti výpadkům. Nicméně také zvyšuje provozní náklady a složitost systému.

V tomto řešení je využito pěti uzlů:

* 1x Config server
* 1x Router (mongos)
* 3x Shard

Tento počet uzlů zajišťuje vysokou dostupnost, redundanci a efektivní rozdělení zátěže.

### Sharding

Sharding je technika horizontálního škálování databází. Umožňuje automatické rozdělení velkého množství dat mezi shardy na základě shard klíče. To může být například unikátní identifikátor, nebo časové razítko. Tento mechanismus pomáhá zvýšit výkon dotazů, neboť každý shard zpracovává část dat, což vede k snížení zatížení jednotlivých uzlů. To implicitně zabraňuje přetížení jednoho serveru, neboť data jsou rozložena podle logického schématu.

V tomto řešení jsou použity tři shardy. To slouží k rovnoměrnému rozložení dat mezi uzly, možnosti budoucího škálování přidáním dalších shardů a redundanci a vysokou dostupnost, neboť každý shard je součástí replikačního setu.

### Replikace

Xxxxxx

### Perzistence dat

Xxxxxx

### Distribuce dat

Xxxxxx

### Zabezpečení

Xxxxxx

# Funkční řešení

Xxxxxx

## Struktura

Xxxxxx

## Instalace

Xxxxxx

# Případy užití a případové studie

Xxxxxx

# Výhody a nevýhody

Xxxxxx

# Další specifika

Xxxxxx

# Data

Xxxxxx

# Dotazy

Xxxxxx

Závěr

Xxxxxx

Zdroje

Xxxxxx

Přílohy

Xxxxxx