Cassandra - Data Model

Tomáš Vlk (vlktoma5@fit.cvut.cz)

May 28, 2019

Úvod

Cassandra je postavená na denormalizaci. Tento přístup vede k absenci operace JOIN a také k datové redundanci[1]. Je součástí takzvané Wide-Column Stores rodiny. To znamená, že se rozrůstá do šířky nikoliv do délky. Pokud se podíváme na nejvíce low-level implementaci, jedná se o key-value storage. Díky kombinaci key-value a Wide-Column Stores je Cassandra "Fancy Hash Table".

Legacy Data model

Původní data model běžel jako Thrift data mode v nových modelech byl nahrazen CQL[2]. I CQL ale pořád převádí kód do původního schématu.

Důležité pojmy

- **Keyspace**
  Udržuje všechny Column Families a replikační faktor. Adekvátní k pojmu "databáze" v relačním světě.
- **Column Family**
  Sdružuje řádky obsahující sloupce. Stejné jako "tabulka" v relační databázi. Jedná se o dvojúrovňové Key-Value úložiště, které nemusí dodržévat žádné schéma. Data přibívají do šířky. Column Family se dá představit jako mapa map. Kde vnější mapa má jako klíč Row Key a vniřní mapa má jako klíč Column Key. Column Key jsou setříděné na lokální uzly. Row Key určují místo na uzlu a nemusí být setříděné.
- **Row**
  Řádka identifikovaná jednoznačným[3] klíčem a obsahuje sloupce s daty. Jednotlivé řádky nemusejí mít stejné sloupce. Podobné jako záznamy v tabulce v relační databázi, ale nemají pevně danou strukturu.
- **Column**
  Sloupec je nejmenší jednotkou v Cassandře. Má svůj název, hodnotu a časové razítko s časem vložení. Podle razítka lze rozhodnout o aktuálnosti záznamu. Sloupce se dále dělí na
  - **Standart** - Standartní obyčejný sloupec, který uchovává jednu hodnotu.
  - **Composite** - Spojený sloupec se používá, pokud je primární klíč složený z více sloupců. Názvy sloupců v takovém případě obsahují svůj původní název rozšířený o druhou část primárního klíče.
  - **Expiring** - Sloupce s omezenou dobou platnosti se hodí, pokud chceme životnost dat omezit nějakou dopředu známou dobou, po kterou jsou data platná. Po vypršení této lhůty jsou data z databáze vymazána.
  - **Counter** - Čítací sloupce můžeme využít, pokud chceme inkrementálně zvyšovat hodnotu v daném sloupci. Tato metoda se však příliš nepoužívá a radějí se data předpočítavají průběžně.

[1] Zapisujeme data ve formátu v jakém je budeme číst
[2] Cassandra query language
[3] Většinou nazíváno primárním

Relational Model	Cassandra Model
Database	Keyspace
Table	Column Family (CF)
Primary key	Row key
Column name	Column name/key
Column value	Column value

Figure 1: Cassandra vs Relační databáze

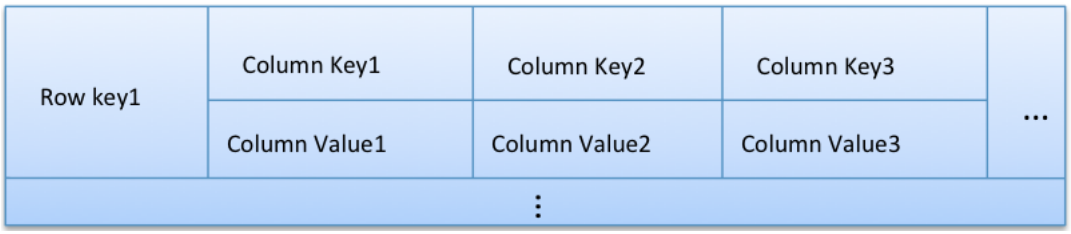

Figure 2: Row model

Replikace

Replikace Column Families se nastavuje na úrovni Keyspace. Z každého Row Key se vytvoří token, který určuje umístění repliky na fyzický uzel.

Existují různé způsoby výpočtu tokenu. Tyto způsoby jsou:

- **Murmur3Partitioner** - Jedná se o počáteční hodnotu, která rozmisťuje data rovnoměrně po clusteru na základě MurMur3 hashovací funkce.
- **RandomPartitioner** - Rovnoměrně umisťuje data po clusteru na základě MD5 hashovací funkce (je pomalejší než MurMur3).
- **ByteOrderedPartitioner** - Ukládá data po clusteru na základě lexikálního pořadí bytů. Nedoporučuje se kvůli složitému vyvažování a nerovnoměrné distribuci dat. Jedinou výhodou je sekvenční hledání podobných klíčů.

Dále existují dva způsoby jak umísťovat další repliky. Ty jsou:

- **Jednoduchá strategie** - Využívá se pouze pro clustery uložené v jednom datovém centru. Tato strategie uloží první repliku na uzel určený rozdělovačem a ostatní repliky jsou uloženy na následujících uzlech v kruhu po směru hodinových ručiček bez ohledu na topologii.
- **Síťová strategie** - Globální replikační faktor se zde změní na replikační faktor pro každé datacentrum. Každé datacentrum má vlastní kruh. Rozmisťování kopií tedy funguje následovně: První kopie se uloží na uzel vybraný rozdělovačem a další kopie ve stejném datacentru se uloží na nejbližší uzel po směru hodinových ručiček, který se nachází v jiném racku.

Modelování

Je vhodné modelovat Cassandra databázy podle dotazů. Nelze rozšířít množství dotazů za pomoci sekundárního indexu jako u relačních databází.