Databázy typu kľúč-hodnota

Zástupca, s ktorým budeme pracovať: Redis

EURÓPSKA ÚNIA
Európsky sociálny fond
Európsky fond regionálneho rozvoja

OPERAČNÝ PROGRAM
ĽUDSKÉ ZDROJE

MINISTERSTVO ŠKOLSTVA, VEDY, VÝSKUMU A ŠPORTU SLOVENSKEJ REPUBLIKY

Tento projekt sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje

www.minedu.sk www.employment.gov.sk/sk/esf/ www.itakademia.sk

Čo je to Redis

- Redis je pamäťová databáza typu kľúč-hodnota
- Môže sa použiť ako cache medzi pomalším diskovým úložiskom a aplikáciou
- Poskytuje PubSub (public-subscribe), skriptovanie, modely uloženia, trasakcie atď
- Podporuje oveľa viac dátových typov ako len String

Prečo použiť Redis?

- Pamäťové databázy sú úžasne rýchle
- Dátové typy používané v Redise sú vhodné na medziprocesovú komunikáciu
  - Cache
  - Rad
  - Riadenie udalostí (publish-subscribe)
  - Rozdeľovanie úloh – load balancing (prúdy)
- Podporuje žurnálovanie a snapshoty na disku
- So servermi sa komunikuje pomocou reťazcových príkazov cez obyčajné sokety – zložité klienty nie sú potrebné

Dátové typy v Redise

- Redis podporuje viacero dátových typov
  - String
  - List
  - Set
  - SortedSet
  - Hash
  - Stream
  - Geo
- Server posiela všetky hodnoty ako reťazce, klienti si ich musia sami konvertovať (textový protokol)

Redis CLI

- Redis má vstavaného konzolového klienta
- Štart: redis-cli
- Ak chcete zistiť či je server aktívny môžete použiť príkaz ping, server odpovie pong.

```
C:\>redis-cli
redis 127.0.0.1:6379> ping
PONG
redis 127.0.0.1:6379>
```

Tutoriál príkazov: http://try.redis.io/

Reťazce/čísla/bloby

- Ľubovoľná postupnosť bajtov (max 512MB)
- Operácie
  - Nastavenie/prečítanie hodnoty: SET/GET
  - Nastavenie ak je nový kľúč: SETNX
  - Reťazcové operácie: APPEND, STRLEN, GETRANGE (získanie podreťazca), SETRANGE
  - Číselné operácie: INCR, INCRBY, DECR, DECRBY
  - Bitové operácie: GETBIT, BITCOUNT, SETBIT

Zoznamy/Listy/Rady/Zásobníky

- Nové záznamy sa pridávajú na začiatok (LPUSH), na koniec (RPUSH), alebo kde chceme (LINSERT, LSET)
- Podobne sa dajú čítať, alebo čítať a hneď mazať zo začiatku (LRANGE 0,0; LPOP) alebo z konca (LRANGE -1,-1; RPOP)
- Maximálny počet prvkov listu je 4 miliardy (2^{32}-1)
- Zložitosť operácií na koncoch listu je štandardne O(1), ale v strede O(n)
- Blokované čítanie a mazanie, pokiaľ nebudú v poli prvky: BLPOP, BRPOP

Množiny

- Neutriedená množina neopakujúcich sa prvkov
- Operácie pridania, mazania, vrátenia v čase O(1)
- Max veľkosť množiny (2^{32}-1) prvkov
- Operácie:
  - Klasické: SADD, SREM, SISMEMBER, SMEMBER (vráti všetky prvky), SCARD (vráti počet prvkov)
  - Množinové: SDIFF, SUNION, SINTER
  - Množinové s uložením výsledku: SDIFFSTORE, SUNIONSTORE, SINTERSTORE
  - Random: SRANDMEMBER – vráti náhodný element, SPOP vráti a zmaže náhodný element

Utriedené množiny

- Množiny neopakujúcich sa prvkov
- Prvky sú utriedené podľa klientom uvedeného skóre a v prípade rovnakého skóre lexikograficky podľa binárnej reprezentácie hodnoty
- Pridanie prvku "ahoj" so skóre 5 a prvku "kvietok" so skóre 2:
  - ZADD moja_množina 5 "ahoj" 2 "kvietok"
- Výpis prvkov v poradí:
  - ZRANGE moja_množina 0 -1
- Inkrementovanie skóre ZINCRBY
- Vrátenie podmnožiny v intervale skóre: ZRANGEBYSCORE
- K dispozícii aj ďalšie operácie ako v množine ...

Hash a.k.a. Hashmapa

- Mapa z blobu do blobu
- Operácie:
  - HSET kľúč hodnota, pre viac naraz: HMSET
  - Čítanie: HGET, HMGET, HKEYS, HVALS, HGETALL (na striedačku kľúč, hodnota)
  - HDEL, HEXISTS, HLEN

Priestorové objekty

- GEO – zoznam geosúradníc s objektom
  - GEOADD kľúč dĺžka šírka objekt
  - GEOPOS kľúč objekt
  - GEODIST kľúč objekt1 objekt2
  - GEORADIUS
    - Vráti utriedenú množinu objektov do danej vzdialenosti v m/km/mi/ft od daného bodu
    - Môže vrátiť aj vzdialenosť (WITHDIST) alebo súradnice (WITHCOORD) týchto objektov
    - Môže sa obmedziť koľko (COUNT) najbližších chceme

Redis ako cache

- Ku kľúču môžeme pripojiť TTL (time to live) cez príkaz:
  - EXPIRE počet_sekúnd
- Ak je databáza nastavená ako cache databáza, záznamy sa zmažú, až keď už nie je v pamäti miesto na vkladanie nových dát

Riadenie udalostí – publish/subscribe

- Klienti sa môžu zaregistrovať na kanály cez SUBSCRIBE
  - Napr. subscribe novinky
- Správa sa pošle do kanála cez PUBLISH, ktorá sa rozpošle všetkým zaregistrovaným
  - Napr. publish novinky "IE mi zaželal pekné vianoce"
- Kanál si nepamätá históriu udalostí
- Klient, ktorý sa zaregistroval, nemôže vykonávať žiadne ďalšie príkazy pokiaľ sa neodhlási (UNSUBSCRIBE)

Prúdy

- Usporiadaná postupnosť „údajov“ defaultne podľa času pridania
  - **V jednoduchom móde** každý comsumer číta postupne všetky údaje, ktoré zatiaľ nečítal nezávisle od ostatných - nemažú sa
  - **V móde skupín**, číta každá skupina všetky údaje nezávisle od ostatných skupín, ale v rámci skupiny comsumer-ov si consumer-i čítajú iba údaje, ktoré si iní členovia skupiny nečítali
- Vidíme aj históriu na rozdiel od publish-subscribe
- Consumer po prevzatí údaja tento údaj „spracuje“ a po spracovaní informuje o ukončení spracovania
  - vieme zistiť, ktoré úlohy neboli potvrdené a v prípade výpadku consumer-a vieme tento údaj priradiť inému členovi skupiny
- Postupnosť údajov sa dá priebežne obmedzovať maximálnym počtom, aby nenarástla na veľmi veľkú

Prúdy - jednoduché

- Operácie:
  - XADD stream id kľúč hodnota, pre viac naraz: HMSET
    - automatické id * vyrába v tvare milisekundy-poradie
  - Čítanie: XRANGE stream - + [COUNT x]
    - - je min id, + je max id
  - Blokované čítanie: XREAD BLOCK timeout STREAMS stream $
    - $ je maximálne id v prúde
    - timeout je v milisekundách, 0 znamená donekonečna
  - XTRIM stream MAXLEN ~ 10
    - Obmedzenie veľkosti prúdu zdola (~ znamená, že môže byť dočasne aj viac, ak je to komplikované pre cluster)

Prúdy so skupinami

- Operácie:
  - Vytvorenie skupiny: XGROUP CREATE stream mygroup $
    - Vytvorí skupinu a nastaví id ktoré už nebude spracovávané skupinou
  - XREADGROUP GROUP mygroup Alice COUNT 1 STREAMS stream >
    - Čítanie členom skupiny mygroup s menom Alice,
    - > je id, ktoré ešte nebolo pridelené žiademu členovi skupiny
  - XACK stream mygroup 1526569495631-0
    - Potvrdenie spracovania údaja s daným id
  - XPENDING stream mygroup - + max_počet
    - Údaje, ktoré boli odoslané členom ale nepotvrdené
  - XCLAIM stream mygroup Peter 3600000 1526569498055-0
    - Člen Peter si vezme údaj s daným ID ak nebol potvrdený predchádzajúcim členom aspoň 3600000 ms = 1 hod od prevzatia
  - XINFO [STREAM | GROUPS | CONSUMERS | HELP]

Transakcie

- **Každý** príkaz v Redise je atomický
- MULTI príkazy EXEC
- Transakcia vykoná príkazy sekvenčne a atomicky
- Ak niektorý príkaz zlyhá kvôli syntaktickej alebo logickej chybe, iba sa preskočí a transakcia pokračuje
- Podporuje optimistické check-and-set operácie pomocou príkazu WATCH.
  - Ak sa sledovaná premenná zmení uprostred trasakcie, transakcia zlyhá.
  - Klient môže, ale aj nemusí zažiadať o opätovné vykonanie.

Perzistencia

- Príkaz SAVE vytvorí fotku (snapshot) aktuálneho stavu databázy na disk
- Príkaz BGSAVE ukladá na disk obsah databázy ako neatomickú operáciu
- Logovanie (voliteľné)
  - Všetky zmeny databázy sa zapisujú do logu na disk (uchovávanie príkazov)
  - Po reštarte databázy sa log znova vykoná
- Logovanie sa začína odznova, ak sa vytvorí fotka

Skriptovanie

- Redis podporuje skriptovací jazyk LUA
- Skripty sa spúšťajú cez príkaz EVAL
- Príkaz EVALSHA hľadá pred tým nahraný skript cez zhodný hash (SHA1)

Replikácia v Redise

- Master-slave replikácia
  - Môže to byť strom závislostí: slave môže byť master pre iný slave
  - Slave-ovia sa aktívne pripájajú na master-a ak padne spojenie
- Replikácia je neblokovaná operácia
  - Master počas nej umožňuje vykonávať ďalšie príkazy
  - Slave môže poskytovať ešte staré dáta

Replikácia v Redise

- Slave dostáva postupne log zmien z master-a
- Slave si môže poprosiť celú kópiu od master-a
  - Master vytvorí snapshot a pošle ho slave-ovi
  - Slave si ho uloží na disk a následne natiahne do pamäte
  - Master pošle log zmien od snapshotu

Delenie dát (partitioning)

- Delenie u klientov
  - Klienti sami určujú kde sú ktoré dáta a pýtajú dáta od toho servera, ktorý ich má
- Delenie cez proxy (napr. Twemproxy)
  - Proxy forwarduje požidavky na ten správny server a aj dodá výsledok
- Redis cluster – preferované riešenie
  - Požiadavka môže odísť na ľubovoľný server a ten mu povie, ktorý server má potrebné dáta (redirecting)

Obmedzenia pri delení dát

- Operácie cez viacero kľúčov, alebo transakcie s viacerými kľúčmi sú nepodporované, ak kľúče nie sú na rovnakom uzle, napr. prienik 2 množín
- Dáta pre jeden kľúč sa nedelia (napr. veľký zoznam)
- Správa dát je náročnejšia

Redis cluster

- Dáta sa delia podľa hash-u do 16384 slotov
- Ak máme 3 uzly uzol A obsahuje napr. sloty 0 až 5500, B sloty 5501 až 11000 a uzol C sloty 11001 až 16383.
  - Ak pridáme uzol D tak sa presunie časť slotov z uzlov A, B a C na D
- Finta: ak chceme, aby určité kľúče boli na rovnakom uzle, môžeme kľúč obohatiť o rovnaký podreťazec medzi kučeravé zátvorky, napr. jano{ahoj} a fero{ahoj},
  - lebo hash sa ráta iba z toho, čo je v zátvorkách { }, ak sa takéto zátvorky v kľúči vyskytujú

Otázky?