NoSQL baze podataka

Predavanje 1: Osnovni pojmovi, tipovi i arhitektura

Univerzitet u Novom Sadu
Fakultet Tehničkih Nauka

Podsetnik

- Slušalite ste i koristoili relacione baze podataka
- Najbolji način da ih se podsetimo, jeste da odgovorite na par pitanja :)
- Kako drugačije?

Lista Pitanja

- Kako rade relacione baze podataka?
- Šta je njihova snaga?
- Na šta se relacioni model oslanja?
- Kako taj model radi?
- Gde su tu nedostaci?
- Koje su tu prednosti?
- Kako rešiti probleme, i možemo li uvke da ih rešimo?

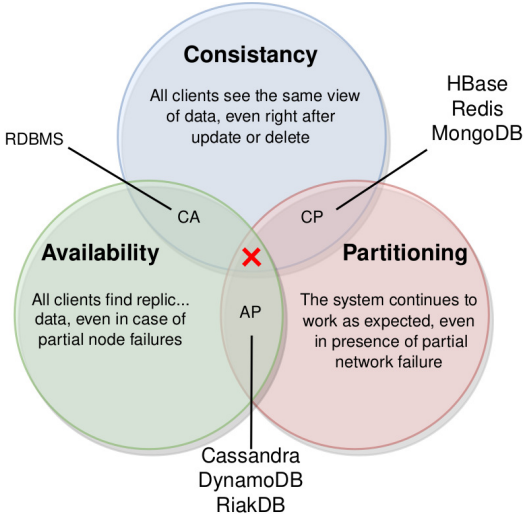
NoSQL baze

- Reprezentuje sve baze podatka čiji model nije relacion po prirodi
- Princip modelovanja, skladištenja podataka i upita se dosta razlikuje od relacionih baza podataka
- Postoje različiti modeli NoSQL baza podataka:
  - Key-value
  - Column
  - Graf
  - Document
  - Time series
  - Mixed models

Upotreba

- Koriste se u raznim situacijama:
  - kada model podataka sa kojim se radi nije relacion po svojoj prirodi
  - radimo sa velikim količinama podataka – ne možemo smestiti na jednu mašinu
  - imamo potrebe za intenzivnim skaliranjem
  - kada je potrebno da samo serijalizujemo/deserijalizujemo nekakv podatak (JSON, XML, itd.)
- Često se koriste u cloud okruženju
- Dosta se koriste u mikroservisnim arhitekturama
- Dosta se koriste u velikim infrastrukturarlnim stvarima – ne uvek!!

- Na važnosti počinju da dobijaju jos u ranim nastancima cloud computing-a
- Relacione baze nisu lako skalabilne u takvom okruženju – moguće je naravno
- Potreba za raznim modelima podatka, ali i velika količina podataka distribuirati na puno čvorova – **CAP** teorema (Eric Brewer)
- Ova teorema kaže da **ne možemo** zadovoljiti sve tri osobine u isto veme:
  - Konzistentnost
  - Dostupnost
  - Particioniranje
- Uvek možemo da dobijemo **najviše** dve od tri osobine
- Odluka se donosi prema potrebama aplikacije – najbolji model za dati problem
- Particioniranje **nije moguće izbeći** u distribuniram sistemima
- Drugi krak biramo spram potreba našeg sistema!


Consistency
All clients see the same view of data, even right after update or delete
HBase
Redis
MongoDB
RDBMS
CA
CP
Availability
All clients find replic... data, even in case of partial node failures
AP
Partitioning
The system continues to work as expected, even in presence of partial network failure
Cassandra
DynamoDB
RiakDB

(CAP teorema)

Sekvencijalni i nasumični I/O

- Ulazno izlazne operacije (I/O) na računaru, nisu jednake u smislu brzine izvršavanja
- Ovde se pod I/O operacijama misli na operacije koje se izvode nad diskom računara
- Nasumično pristupanje podacima je mnogo sporije i manje efikasno od sekvencijalnog (uzastopnog) pristupa
- Brže je pisati/čitati iste podatke sa jednim uzastopnim I/O, nego više manjih nasumičnih I/O operacija po disku
- Sekvencijalne I/O operacije su preferirane od strane NoSQL baza – lakše možemo čitati i pisati velike količine podatka
- Nasumične I/O operacije se češće nalaze kod relacionih baza

Strukture podataka na disku

- Strukture podataka koje koristimo prilikom čuvanja podatka na disk igraju bitnu ulogu u budućem radu sistema
- Ovde imamo dve mogućnosti:
  1. **Promenljive (mutable) stukture**
     - Alociranje memorije
     - Izmena dela podatka (in place)
     - Fragmentiacija prostora
     - Uglavnom nasumičan I/O
  2. **Nepromenljive (immutable) strukture**
     - Nema izmena podatka nakon zapisa
     - Čitanje podatka iz više izvora (fajlova)
     - Zahteva operaciju spajanja (merge)
     - Uglavnom sekvencijalni I/O
     - Lako za izmeniti model

- NoSQL baze podatka uglavnom koriste nepromenljive strukture podataka
- Ove strukture su optimizovane za operaciju zapisivanja
- Želimo da efikasno zapišemo potencijalno veliku količinu podataka
- Želimo da brzo zapišemo podatke
- Želimo da efikano repliciramo podatke na druge čvorove u sistemu
- Želimo da vržimo sekvencijalne I/O operacije za sve funkcije našeg sistema
- Želimo da izbegnemo nasumičan I/O što je više moguće
- Pogotovo ako radimo sa velikim količinama podataka

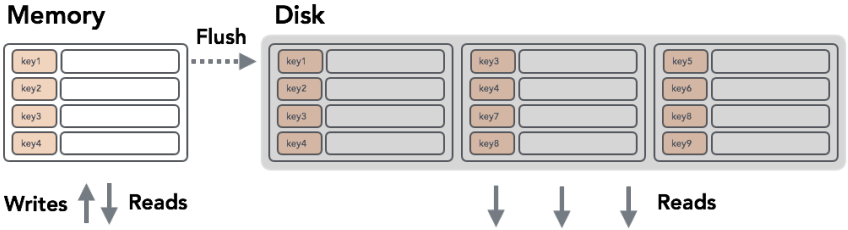
Pitanje 1

Ako znamo prethodno, kako onda relacione baze čitaju podatke sa diska?

Pitanje 2

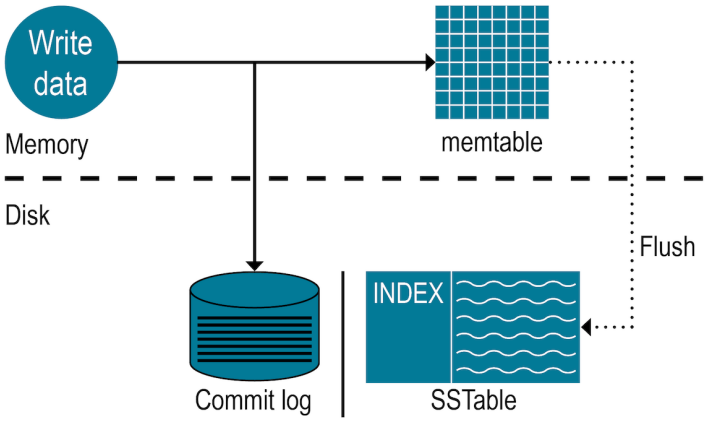
Koji tu problem potencijalno imamo?

Sekvencijalni zapisi



(Databases Internals)

Zapis podataka



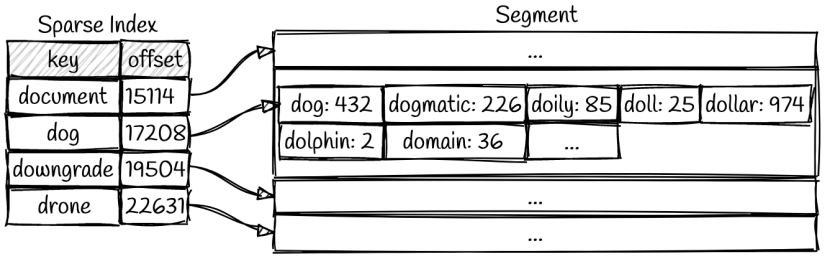
(Cassandra write path)

Write Ahead Log - Buffered i Unbuffered I/O

- Ovo je jedna od velikih tema za debatu u dizajnu baze podataka — baferovan nasuprot nebaferovan I/O
- Danas aplikacije zahtevaju više od baza podataka, dok diskovi ne drže korak sa ovim zahtevima
- Da bi se diskovi učinili bržim, OS mapira delove diska u memoriju (tema za sledeći put)
- Ovaj mehanizam amortizuje razlike u brzinama diskova i memorije

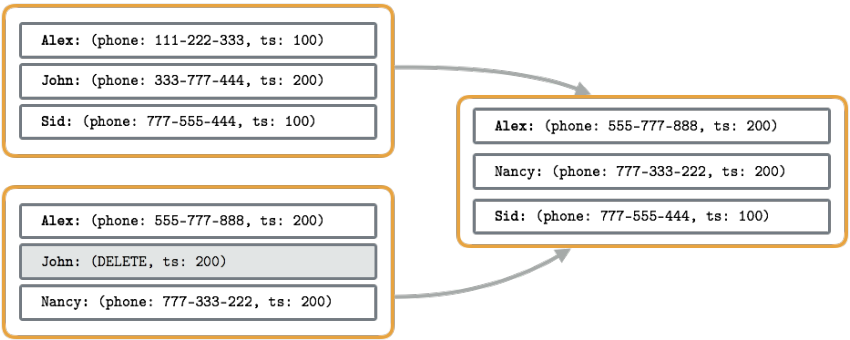
- Promene na disku se dešavaju samo u memoriji, a periodično OS upisuje promene na fizički disk
- Ovo je poznato kao baferovani I/O — zapisujemo podatke u bafer koji se na kraju isprazni na disk – kada se za to stvore uslovi
- Baferovani I/O se može izbecći korišćenjem I/O bez baferovanja — podatke upisujemo direktno na fizički disk odmah kako dolaze
- Ovo može rezultovati prevelikim brojem operacija ka disku
- Dodatno usporava sistem, ali daje stroge garancije trajnosti podataka

Tabele sortiranih stringova – SSTable



(LSM — Write Heavy Databases)

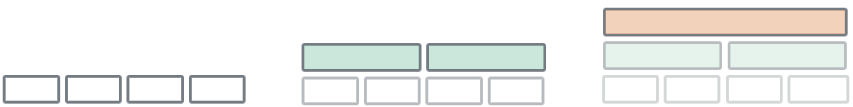
Proces spajanja



(LSM Tree compaction)

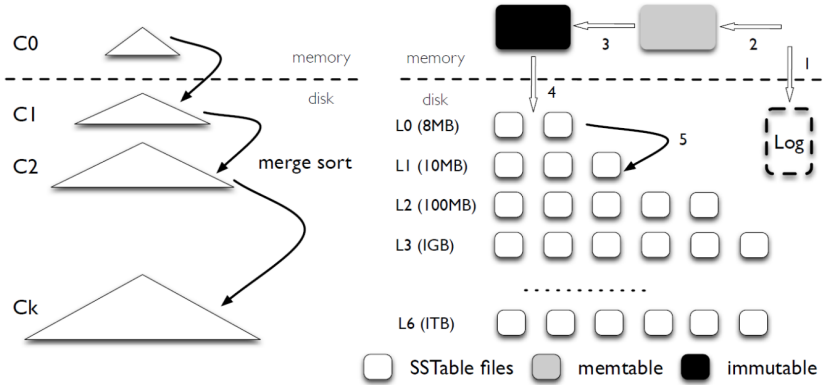
Kompakcije

- Oslobadjanje zauzetog prostora



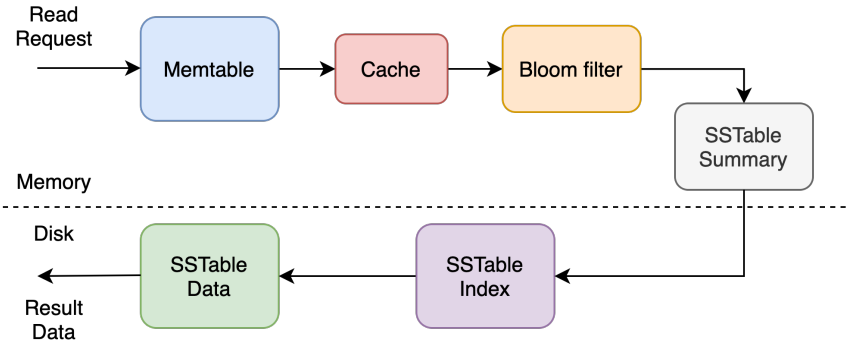
(LSM Tree compaction)

LSM Stabla



(LSM Tree compaction)

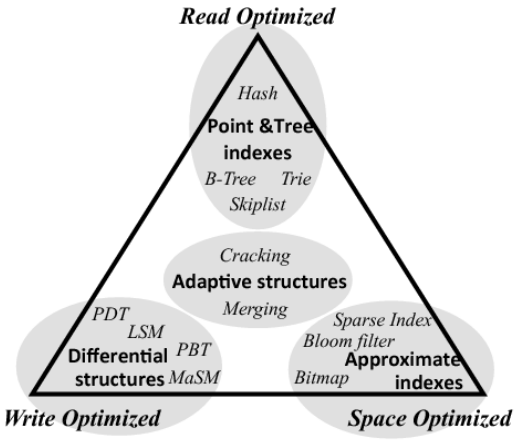
Čitanje podataka



(Cassandra write path)

Metode pristupa – RUM prepostavke

- Definisano od strane istraživača sa Harvarda
- Tri ključna aspekta za šta baza može biti optimizovana
- Čitanje **R**, Pisanje/Izmena **U**, Memorija/Prostor **M**)
- Uzimamo u razmatranje kada biramo bazu podataka
- Uzimamo u obzir kada konsultujemo domen problema



(Designing Access Methods: The RUM Conjecture)

Dodatni materijali

- Seven Databases in Seven Weeks: A Guide to Modern Databases and the NoSQL Movement
- Database Internals: A Deep Dive into How Distributed Data Systems Work

Pitanja

Pitanja :) ?