NoSQL baze podataka

Predavanje 4: Konzistentnost, ACID, BASE, Agregiranje podataka

Univerzitet u Novom Sadu
Fakultet Tehničkih Nauka

Uvod

- Već smo pominjali CAP teoremu
- Ona nam kaže da u distribuiranom okružnju ne možemo zadovoljiti sve tri osobine
- Najviše možemo imati 2, od čega particioniranje ne možemo izbeći
- Sa druge strane, ako koristimo relacione baze ne možemo tako lagano skalirati sistem
- Ali imamo garanciju za konzistentnost

NoSQL - Eventual Consistency

- Zbog distribuiranog modela, svaki server može odgovoriti na svaki upit
- Serveri medjusobno komuniciraju “iza scene”
- Moguće je da server koji odgovara na upit nema najsvežije podatke.
- Kod nekih primena to ne mora biti problem – da li se istog momenta vidi poslednji pristigli tweet.
- Kod primena u radu s dobro struktuiraim podacima gde se zahteva stroga konzistentnost ovo je problem.

ACID

- **ACID** je skup osobina koje garantuju pravilno izvršavanje transakcija (definisan krajem 1970-ih godina):
  - **A**tomicity (atomarnost)
  - **C**onsistency (konzistentnost)
  - **I**solation (izolovanost)
  - **D**urability (trajnost)

- **A**tomicity (atomarnost) – Transakcija kao logička jedinica posla mora predstavljati atomarni skup aktvnosti koji se izvršava u celosti ili se poništavaju njena dejstva
- **C**onsistency (kozistentnost) – obezbedjuje da baza podataka konzistentno promeni svoje stanje posle uspešnog potvrdjenih (comited) promena
- **I**solation (izolacija) – Transakcija se izvršava izolovano, ne utiče na rad ostalih transakcija nad bazom podataka koje se "paralelno" izvršavaju
- **D**urability (trajnost) – obezbedjuje da potvrdjeni (comited) rezultatai ili efekti transakcije budu trajni

SUBP i CAP

- Većina relacionih SUBP inicijalno su bili projektovani da se izvršavaju na samostalnim serverima
- P iz CAP teoreme je bez značaja, jer je baza podataka na jednom serveru
- Server je u funkciji i radi ili ne radi
- Ne može se govoriti o situacijama da je server delimično raspoloživ
- Kod relacionih SUBP pažnja se fokusira na konzistentnost

NoSQL i CAP

- Kod pojedinih NoSQL baza podataka pažnja se fokusira na C i A deo CAP teoreme
- Pri projektovanju NoSQL baza u startu se pošlo od činjenice da é se izvršavati na:
  - desetinama čvorova,
  - stotinama čvorova,
  - pa čak i na hiljadama čvorova nekog Data Center-a
- Tolerancija na otkaze postiže se tako što se baza podataka kompletno replicira na više data center-a.

- Prednost sistema baza podataka koji relativno jednostavno podržavaju replikaciju u odnosu na tradicionalne relacione sisteme je:
  - Zadatak koji treba obaviti rasporedjuje se na sve računare u sistemu
  - Postiže se veoma dobro vreme odziva čak i kod velikog broja čitanja i pisanja u bazu podataka.
- Kod NoSQL baza podataka se implementira takozvana eventualna (ili konvergentna) konzistentnost – evetual consistency
- Promene u bazi se ne repliciraju istovremeno — replicated eventualy
- Pojedini čvorovi ili grupe čvorova nemaju u svakom trenutku poslednje stanje podataka.

- Ponekad se ide u pravcu popuštanja u konzistentnosti organizacije podataka;
- U korist skalabilnosti i postizanja boljeg vremena odziva celog sistema.
- Odbacivanje konzistentnosti:
  - Obezbedjuje se raspoloživost i tolerancija razdvojenosti;
  - Ne garantuje se čitanje poslednje verzije podataka u slučaju razdvojenosti;
  - U suprotnosti s ACID osobinama

NoSQL i BASE

- Umesto **ACID** osobina definiše se **BASE** skup osobina;
- **B**asically **A**vailable, **S**oft state, **E**ventual Consistent.
- Definisao ih Eric Brewer tvorac CAP teoreme;
- BASE skup osobina predstavlja suprotnost ACID skupu osobina u cilju postizanja kompromisa eliminacijom konzistentnosti;
- Obezbedjuje se dostupnost podataka i tolerancija razdvojenosti.

- **Basically Available** – Suštinski raspoloživ — sistem obezbedjuje dostupnost većini podataka, većina podataka je dostupna veći deo vremena
- **Soft state** - Nekonzistentno stanje - baza podataka ne mora biti konzistentna u svakom trenutku, stanje sistema menja se s vremenom, čak i kada nema unosa podataka.
- **Eventually consistent** – Konvergentna konzistentnost Sistem ne garantuje da će svi čvorovi sistema sadržati iste kopije podataka, teži se vremenskoj tački kada će svi čvorovi sadržati konzistentne podatke.

- Sledeći način na koji se NoSQL baze podataka udaljavaju od tradicionalnog načina rada s podacima je napuštanje relacionog modela.
- Neki podaci prirodno ne odgovaraju korišćenju relacionog modela podataka iz različitih razloga
- Zbog toga što:
  - Podaci često menjaju formu;
  - Dužinu podataka
  - Zbog toga što su potpuno nestruktuirani

Cloud computing

- Postavlja se pitanje žasto je to tako i šta je do toga dovelo?
- Danas je posebno aktuelan Cloud computing koncept:
  - Resursi se korisnicima isporučuju u vidu usluga preko mreže
  - Najčešće preko interneta;
  - Sami fizički resursi su nezavisni od lokacije.
- Svaki resurs može da bude dostupan kao servis korisnicima
- Plaćanje po utrošku – pay as you go

- Po svojoj prirodi cloud computing je jedan veliki (centralizovan) distribuiran sistem
- Termin distribuirani sistemi se orginalno, u početku, odnosio na računarske mreže u kojima su pojedini računari bili prostorno distribuirani.
- Danas se o distribuciji govori na mogo širi način;
- I kada se radi o autonomnim procesima koji se izvršavaju na jednom fizičkom računaru.
- U medjusobnoj interakciji su razmenom poruka.

Distribuirano računarstvo – osnove

- Distribuirani softverski sistemi su prilično složeni za modelovanje iimplementaciju
- U ovakvim sistemima često nastajezbog problema u komunikaciji čvorova preko mreže koja nije sigurna i pouzdana
- Poruke mogu da kasne, mogu da stignu u različitom redosledu ili da ne stignu
- Takodjee, čvorovi u sistemu mogu prestati sa radom potpuno nasumično stvarajući dodatne komplikacije
- James Gosling i Peter Deutsch, kreirali su listu problema za mrežne aplikacijepoznate kao 8 zabluda distribuiranih sistema

8 zabluda distribuiranih sistema

1. Mreža je pouzdana
2. Kašnjenje ne postoji
3. Propusnost je beskonačna
4. Mreža je sigurna
5. Topologija se ne menja
6. Postoji samo jedan administrator
7. Troškovi transporta ne postoje
8. Mreža je homogena

- Iako razvoj distribuiranih sistema traje već nekoliko decenija, problemi koji se javljaju prilikom njihovog razvoja su i dalje identični.

Osnovne osobine

- Nekoliko entiteta učestvuje u komunikaciji, ali, u isto vreme mogu obavljati i nezavisne operacije
- česnik u komunikaciji je hardverska komponenta ili softverski proces
- Klijenti imaju utisak da komuniciraju sa jednim čvorom, to znači da čvorovi u sistemu moraju saradjivati/komunicirati da bi ostvarili nekakv cilj
- Tri osnovne karakteristike:
  1. Konkurencija – više aktivnosti mogu da se izvršavaju, ali i na više čvorova
  2. Nezavisni otkazi – čvorovi otkazuju potpino nezavisno jedan od drugog
  3. **Nepostojanje globalnog sata** – svaki čvor ima svoj tok vremena

Osobine čvorova

- Čvorovi u distriuiranom sistemu mogu da imaju razne uloge
- U zavisnosti od namene od okruženja u kom se nalaze mogu da izvršavaju drugacije operacije
- Dve osnovne uloge su:
  1. Procesiranje – obrada podataka
  2. Skladištenje podataka
- Čvor može da ima i kombinovanu ulogu u nekim slučajevima – volonterski čvorovi
- Kod proceranja podataka treba obraditi pažnju na jedan zanimljiv element – **loklanost podataka** (Big Data)
- Ideja: Prebaciti proračune ka podacima umesto obratno – skupa operacija pogotovo na udaljeno mesto

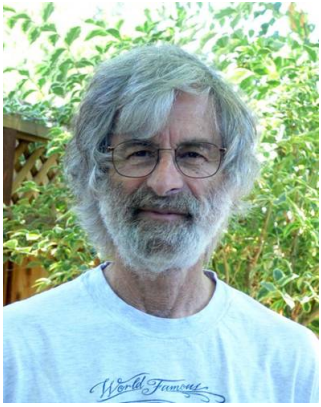
Protokoli komunikacije

- Ovakvi sistemi po sovojo prirodi se oslanjaju na komunikaciju čvorova
- Čvorovi izmedju sebe izvršavaju razne protokole da bi obavili čak i osnovne operacije
- Ovih protokola ima nekoliko, i pokušavaju da emuliraju operativni sistem u nekoj meri
- Ovi protokoli nisu nužno jednostavni, i za svoju ispravnost uglanvom se oslanjaju na dosta matematike
- Obično testiranje *ovde nema nikakvu moć* :)

Zašto nema moć?

If there is a bug in a *distributed algorithm*,
no matter how improbable it may seem,
it's not a question of **whether** it will appear,
it's a question of **when** it will appear.

(Leslie Lamport, Heidelberg Laureate Forum 2021,
https://www.youtube.com/watch?v=KVs3YFKqclU)



(Leslie Lamport, Turing award amongst others)

- Zbog svih ovih osobina, postavlja se pitanje da li je kraj relacionih baza podataka?
- Odgovor je NE
- Nisu svi problemi relacioni po prirodi, niti su nerelacioni
- Sa druge strane, ako nemamo previše podataka relacione baze će uraditi dobar posao
- Imamo razne alate i kada smo u dilemi šta da koristimo, konsultujemo domen problema

Uvod

- Jedan od problema koji se javljaju sa eventulanom konzistenciom i velikim brojem čvorova jeste svakako i razrešenje konflikta
- Pored toga kako obebediti strožiju konzistentnost, sa dobrim osobinama za skaliranje
- 2011 Marc Shapiro i saradnici, objavili su rad na temu jake eventualne konzistenstnosti – strong eventual consistency
- Gde sistem, u smislu konzistencije, se nalazi izmedju dva ekstrema
- Na taj način imamo jače garancije konsitentnosti i mogućnost skaliranja

- Ovo su omogućili tipovi podataka sa automatskim razrešenjem konfliksta – conflict free replicated data types (CRDTs)
- Korisnici moraju da pišu svoje sisteme tako da koriste ove specifične tipove podataka da bi dobili prethodne osobine
- Dokazano je da svi JSON tipovi, i proizvoljna kompozicija istih, može da se iskoristi bez bojazni da će model biti narušen
- Ovo nam znatno olakšava posao!

- ALI i dalje moramo da razumemo kako ovi tipovi rade, i kako da ih koristimo
- Nezgodna stvar je da je ova tehnologija reativno nova – mali broj dostpounih biblioteka
- AKo nešto treba da uradimo uglavnom moramo sami da implementiramo – nije strašno
- Neki sistemi već koriste ovaj mehanizam (Riak, AntidoteDB) i nude tipove podataka koje korisnici mogu da koriste
- Drugi sistemi koriste ovaj mehanizam za repliakciju podatka izmedju data centara (Redis)

Uvod

- Modelovanje podataka u NoSQLbazama je dosta različito od onog u relacionim bazama
- Pre svega, prva bitna stvar je poznavnje upita
- Unapred moramo poznavati upite da bi znali kako da sačuvamo podatke na najbolji mogući način
- Ovo je pre svega zbog načina kako su podaci skladišteni, ali i načina skliranja

- Agregate (agregacija) je skup objekata istog domena koji se mogu tretirati kao jedna celina.
- Primer u poslovnom informacionom sistemu može biti zaglavlje fakture i stavke, koje se tretiraju kao posebni objekti, ali je zgodno rukovati fakturom (zaglavlje + stavke) kao jednom celinom – agregacijom.
- Jedana od komponenti agregiranog objekta je koren agregacije.
- Referenciranje spolja se može se vršiti samo preko korena agregacije.

- Na taj način koren može obezbediti integritet agregacije kao celine
- Agregacija je osnovni element prenosa izmedju aplikacije i podsistema diskova, odnosno baze podataka.
- Pojam agregacija je čest i koristi se u različitim kontekstima, pri čemu se ne odnosi uvek na isti koncept kao kod Domen-Driven Design Agregate.
- Kada su u pitanju baze podataka, posebno pojavom NoSQL baza, osnovna ideja je bila da se podaci na efikasan način smeštaju na klastere većeg broja računara
- Skladištenje agregiranih objekata u bazu podataka kao osnovne jedinice prenosa daje puni smisao ideji da se sistem baze poddataka izvršava na klasteru računara.

- Agregacije predstavljaju prirodne jedinice za različite strategije distribucije kao što je Sharding, jer postoji relativno veliki skup podataka kojem se pristupa kao celini.
- Koncept agregiranja u pristupu podacima ima smisla i s programerskog aspekta.
- Kada se koriste agregirani koncepti, kao kod NoSQL baza podataka, preslikavanje sa ekrana/forme je mnogo jednostavnije, jer se kompletan sadržaj koji se preuzima može smestiti u bazu podataka kao jedna celina.

Problemi

- Sve radi veoma dobro kada se podacima pristupa preko agregacije kao celinie.
- Šta ako se želi drugačiji pogled na podatake i podacima je potrebno pristupiti na drugačiji način?
- Zbog toga unapred moramo poznavati naše upite
- Ili podatke čivati u nekoliko oblika da bi obezbedili razne varijante čitanja

Dodatni materijali

- A Low Overhead High Performance Buffer Management Replacement Algorithm
- Policy with Applications to Video Streaming
- https://hazelcast.com/glossary/cache-miss/
- Data Caching in Cassandra

Pitanja

Pitanja :) ?