Univerzitet u Nišu
Elektronski fakultet
Katedra za računarstvo

1. seminarski iz predmeta
Sistemi za upravljanje bazama podataka

Distribuirana arhitektura Apache Cassandra skladista podataka

Mentor:

Prof.dr Aleksandar Stanimirović

Student:

Teodora Stefanović 1296

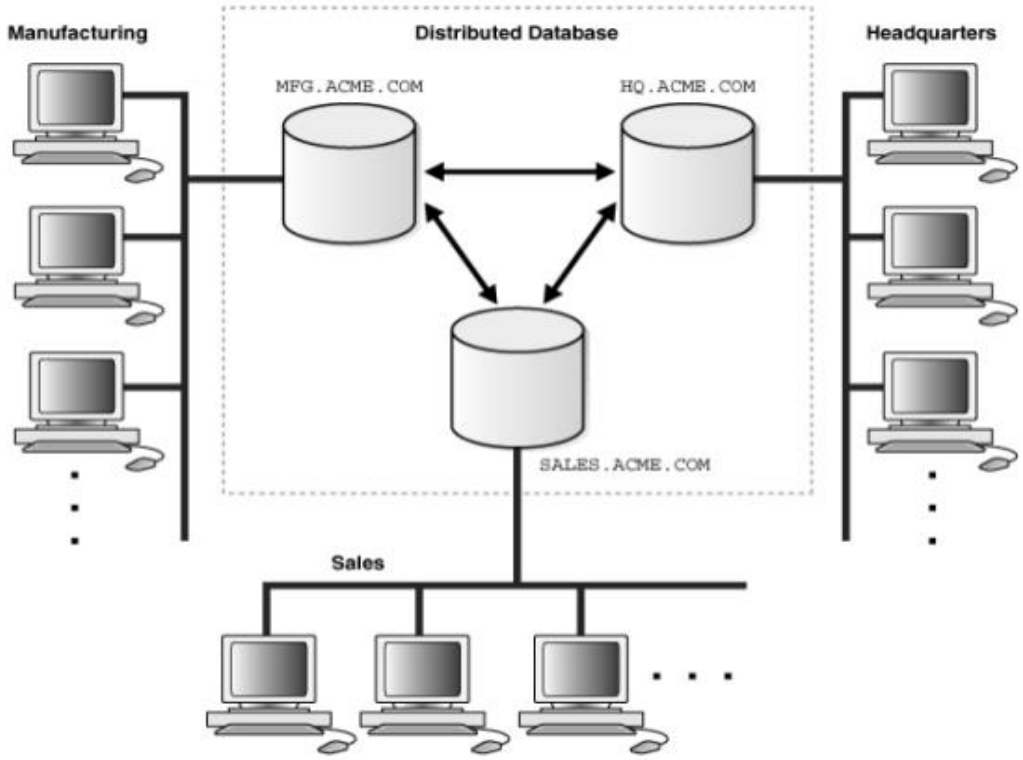
Niš, 2022.

Sadrzaj

1. Opis problema

Baza podataka koja je podeljena na više fizičkih čvorova koji se nalaze na različitim lokacijama je distribuirana baza podataka. Čvorovi distribuirane baze podataka su povezani komunikacionom mrežom.

Podacima se mogu pristupiti kao da se nalaze na jednom čvoru jer je glavni cilj da se ne primete pristupi različitim čvorovima. Svaki čvor ima sistem za pristup bazi podataka ili DBMS (Database management system). Distribuirana baza podataka, prikazana je na slici:



1.1. Apache Cassandra

Apache Cassandra je otvorenog koda, distribuiran i decentraliziran sastav za skladiste podataka te se koristi za upravljanje vrlo velikim količinama podataka raširenih sveta. Apache Cassandra je:

- Skalabilna, otporna na greške i dosljedna
- Ključ-vrijednost i kolona-orijentirana baza podataka
- Distribucijski dizajn temeljen je na AmazonovomDynamo i GoogleovomBigTable modelu podataka
- Stvorio ga je Facebook te joj se sastav upravljanja značajno razlikuje od relacijskih baza podataka
- Implementira Dynamov stil replikacionog modela koji je otporan na greške, ali nadodaje model podataka „porodica kolona“ (columnfamily)

Zbog svojih izvrsnih tehničkih osobina, Cassandra je postala vrlo popularna, a neka od tih osobina su:

- Elastična skalabilnost – Cassandra dopušta dodavanje hardvera kako bi se zadovoljile potrebe korisnika i njihovim sve većim zahtevima za podacima.
- Brze i linearno proporcionalne performanse – Cassandra je linearno skalabilna, što znači da se povećanjem broja čvorova, povećava i propusnost podatak. Dakle, odrzava brzo vrieme odziva.
- Prilagodivo skladište podataka – Cassandra podrzava sve moguće oblike podataka, strukturirane, polu-strukturirane i nestrukturirane. Ona se dinamički prilagodi promenama strukture podataka prema korisnikovim potrebama.

• Jednostavna distribucija podataka – Cassandra omogućuje prilagodljiv način distribuiranja podataka repliciranjem tih podataka preko mnogih centara podataka.

• Podrzavanje transakcije–Cassandra podrzava transakcije, ali ne omogućava nuzno svojstva nedeljivosti (eng. Atomicity), doslednosti (eng. Consistency), izolacije (eng. Isolation) i izdrzljivosti (eng. Durability)

• Brzo pisanje – Cassandra je osmišljena za pokretanje na jeftinom hardveru. Ona izvršava jako brza pisanja i moze uskladištiti stotine terabajta podataka bez zrtvovanja efikasnosti čitanja

Cassandra ima eng. peer-to-peer distribuirani sastav preko svojih čvorova i podatak je distribuiran preko svih čvorova u klasterima. Svi čvorovi u klasterima imaju jednaku ulogu te je svaki čvor nezavisan i u isto vrieme medjusobno povezan s drugim čvorovima.

Takodje, svaki čvor u klister moze prihvatiti zahteve za pisanje i čitanje bez obzira na mesto podatak u klasteru.
Kada jedan čvor ispadne iz klastera, zahtevi za pisanje i čitanje obradjuje drugi čvor u mrezi.

Svi podaci koje sadrži jedna instanca Cassandre deo su klastera (cluster), odnosno prstena (ring).
Prsten je podeljen na više prostora kljuceva koji imaju svoje ime i za koje se definisu konfiguracioni parametri poput strategije replikacije.

Prostor kljuceva otprilike odgovara jednoj bazi podataka u relacijskom modelu.
Prostori kljuceva sadrže tablice, sortirane kolekcije redova koje su podeljene na particije.

Na nivou tablice definisu se kolone koje redovi tablice sadrže, ali se ne zahteva da svaki red ima svaki kolonu, za razliku od relacijskih baza podataka gde svaka kolona ima vriednost, iako ona može biti posebna vriednost koja oznacava nedostajucu ili nepoznatu vrednost.

Na razini tablice takoder se defise koji podskup kolona cini primarni kljuc.
Kolone primarnog kljuca dele se na kljuc za particionisanje, kljuc za grupisanje te staticne kolone.

Kolone kljuca za particioniranje služe odredjivanju particije kojoj red pripada.

Unutar jedne particije, redovi su sortirani prema vrednostima kolona koji cine kljuc za grupisanje.

Staticni stupci su stupci koji su jednaki za sve redove koji se nalaze unutar iste particije pa se oni spremaju samo na jednom mestu, umesto da se sprema ista vriednost za svaki red.

Kolona se sastoji se od imena kolone i njegove vrednosti, koja mora biti tipa koji je definisan za tu kolonu na 16. nivou tablice. Osim toga, svaka kolona sadrži i dve dodatne informacije, vreme kad je vrednost zapisana te vreme isteka. Vreme zapisivanja služi razrešavanju konflikata, na nacin da se najnovija vrednost uvek smatra ispravnom.

Vreme isteka, koje ne mora biti postavljeno, oznacava kad kolona istice, odnosno do kojeg je trenutka vrednost te kolona moguce citati.

2. Glavni delovi Cassandre su:

• Čvor (Node) – mesto gde su podaci spremljeni.

• Centar podataka (Data center) – kolekcija povezanih čvorova.

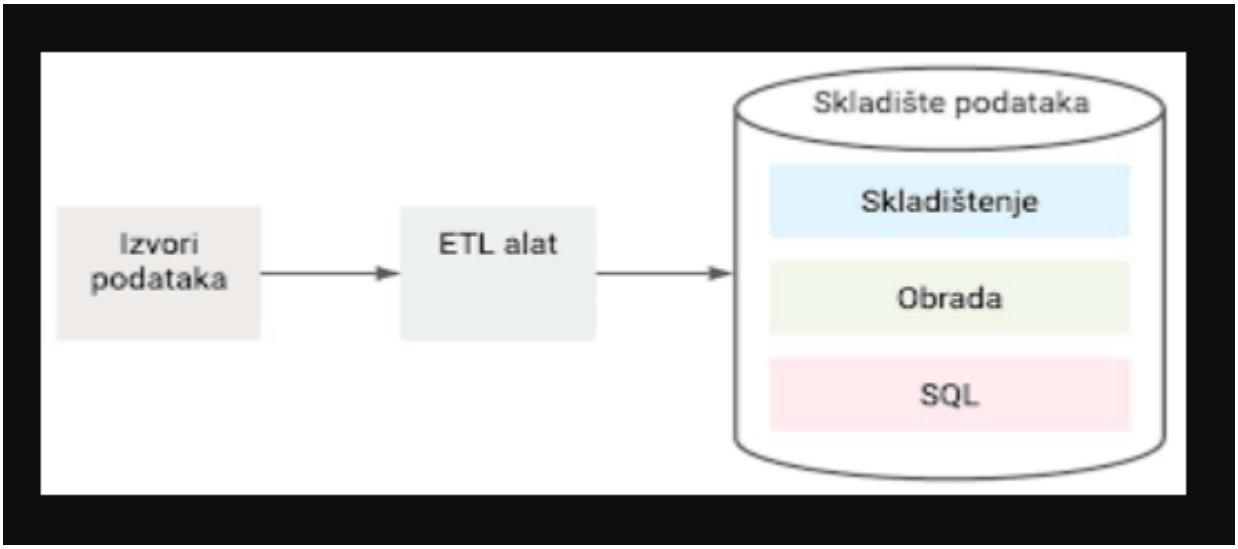
• Klaster (Cluster)–komponenta koja sadrzi jedan ili više središnjica podataka.

• Log izvršenja (Commit log) –mehanizam za oporavak od pada sistema, svaka operacija zapisivanja je zapisana u log izvršenja.

• Memorijska tablica (Mem-table) – tablica u kojoj su smeštene strukture podataka. Nakon dnevnika izvršenja, podatak će biti zapisan u memorijsku tablicu. Ponekad će biti višestrukih memorijskih tablica za jednu kolonu.

• SSTablica (SortedStructure Table) –datoteka na disku u koju je podatak prebačen iz memorijske tablice kada njen sadrzaj dosegne najvišu dopuštenu vriednost.

• Bloom filter–brz, neodredjeni algoritam za proveru da li je element deo nekog kompleta. On je posebna vrsta predmemorije kojoj se pristupa posle svakog upita



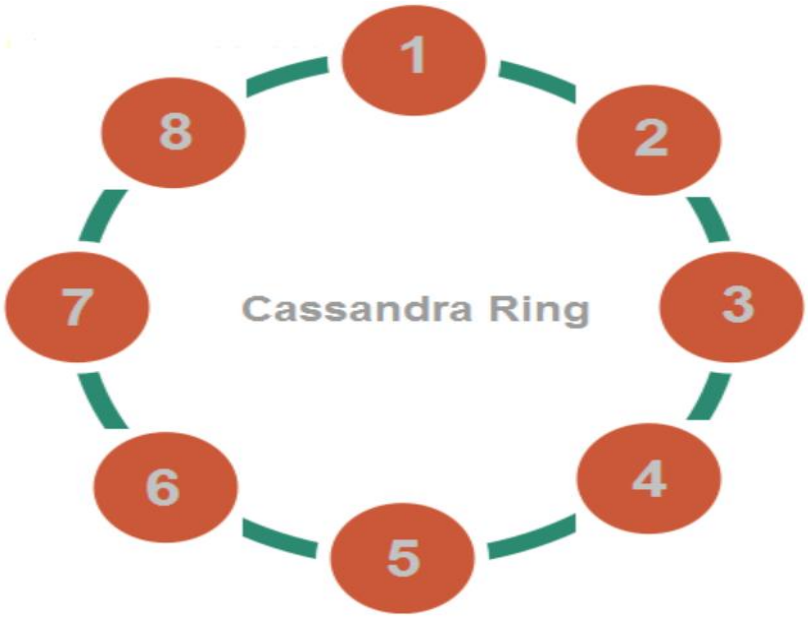
3. Prikaz prstena Raspodela zetona unutar akumulacije prikazuje oblik prstena

Delitelj (partitioner) generise zetone od 0 do 255 i u klasteru je 5 Cassandra masine. Kako bi se dodelio jednak teret, svaki čvor trebao bi nositi jednak broj zetona.

Dakle, prvi čvor bit će odgovoran za zetone od 1 do 51, drugi će posedovati zetone od 52 do 102, treći od 103 do 153, četvrti od 154 do 204 i peti od 205 do 255.

Ako se svaki čvor označi s brojem zetona kojeg moze posedovati, onda akumulacija dobija izgled prstena. Delitelj je hash funkcija koja odredjuje opseg mogućih ključeva redova (rowkey, row ID).

Pri konfiguriranju Cassandre, pozeljno je postaviti najveći broj zetona koje neki čvor moze posedovati, ali treba imati na umu da dodavanjem novih čvorova ili odbacivanjem starih čvorova, jednaka podela zetona moze biti neuravnotezena.



4. Zapisivanje podataka

Klijenti trebaju biti spojeni na bilo koji Cassandrin čvor i poslati zahtev za pisanje ako zele zapisati neki podatak.
Taj čvor se naziva koordinacijski čvor (eng. coordinatornode) i on je mozda pravo mesto za zapisivanje tog podatka.

Kada čvor u klasteru primi zahtev za pisanje, tada taj zahtjev obradjuje uslugaStorageProxy.
Zadatak StorageProxy-a je dohvatiti sve čvorove (sve kopije) koje su odgovorne za drzanje podatka koji će biti zapisan te optimizvati strategiju kopiranja.

Kada su čvorovi za kopiranje identifikovani, on šalje RowMutation poruku i čeka odgovore, ali ne čeka sve odgovore već samo onoliko koliko je definisano u ConsistencyLevel opciji (dovoljno da zadovolji korisnikov najmanji broj uspešnih zapisa).

Nakon toga ima tri toka odvijanja radnje:

4.1. FailureDetector

Otkriva ima li dovoljno čvorova za zadovoljavanje ConsistencyLevel-a. Ako nema, zahtev za pisanje se ne izvršava.

4.2. Hintedhandoff

Ako FailureDetector detektira dovoljno čvorova, ali ispiše pauzu (time-out) posle zahteva zbog problema u infrastrukturi ili prevelikog opterećenja, StorageProxy 6 lokalnih savet-a (hint) za ponovno zapisivanje kada se podbačeni čvorovi vrate nazad I umreze. Taj postupak se naziva isporuceni savet (hintedhandoff)

4.3. Zapisivanje

Ako je sve uredu, izvršava se zapisivanje podatka.

Ako su čvorovi koji primaju kopiju, distribuirani preko sredine podataka, preporučljivo je slati poruke jednoj kopiji u svakoj sredini podataka sa zaglavljem koji naredjuje prosledjivanje zahteva drugim čvorovima koji primaju kopiju u toj okolini podataka.

Unutar čvora, prvo je podatak dodat kao prilog u dnevniku izvršenja (Commit log) te je prosledjen memorijskoj tablici (Mem-table) u odgovarajuću porodicu kolone u memoriji.
Kada se memorijska tablica napuni, njeni podaci se premeste (flush) na disk u sortiranu strukturu koja se naziva SSTablica. S mnogo premeštanja (flush), disk dobija puno SSTablica.
Za upravljanje tim tablicama, odvija se proces zbijanja (compactionprocess). Taj proces spaja podatke iz malih SSTablica u jednu veliku i sortiranu datoteku.

5. Čitanje podataka

Kada StorageProxy čvora na koji je klijent spojen, dobije zahtev za čitanje, on dobije popis čvorova koji sadrze taj ključ temeljeno na strategiji kopiranja (ReplicationStrategy).

Tada StorageProxy sortira čvorove prema udaljenosti od sebe.
Ta udaljenost odredjuje se Snitch funkcijom koja je postavljena za tu akumulaciju.
Nakon nabavljanja zeljenih ključeva dolazi čitanje podatka.
Koordinatorski čvor (onaj na kojeg je korisnik spojen) šalje naredbu za čitanje najblizem čvoru i vraća podatak.

Na osnovi ConsistencyLevel-a, drugi čvorovi će poslati naredbu za izvršavanje operacije čitanja i poslati skraćen pregled rezultata. Ako je omogućena opcija ReadRepair, ostalim čvorovi koji sadrze kopiju bit će poslana poruka za izračunavanje skraćenog pregleda rezultata.

Taj zahtev se izvršava u pozadini, nakon što su vraćeni rezultati. Zahvaljujući tome, svaki čvor je azuriran, što kopije čini doslednima.

Unutar čvora, podatak se pretrazuje u memorijskoj tablici i ta je pretraga jako brza zato što postoji samo jedna kopija zeljenog podatka.

Ako se podatak ne nalazi u memorijskoj tablici, pretrazuje se SSTablica.

Pre je spomenuto da svakim premeštanjem (flush) podataka iz memorijske tablice, stvara se nova SSTablica te tako postoji nekoliko SSTablica koje mogu sadrzavati zeljeni podatak.
Zato je svaka SSTablica povezana sa svojim Bloom Filterom za ključeve redova koji je u memoriji i koristi se za otkrivanje zeljenog podatka u odredjenoj SSTablici.

Podaci u SSTablici su sortirani hronološki od najnovijeg podatka do najstarijeg podatka.

Cassandra pretrazuje podatke od najnovijeg do najstarijeg i ako naidje na zeljeni podatak, korisniku vrati vriednost i prestaje s pretrazivanjem jer korisniku je potrebna najnovija vrijednost. Osim Bloom Filtera za ključeve redova, postoji Bloom Filter za svaki red SSTablice.

Drugi Bloom Filter otkriva postoji li zeljeno ime kolone u SSTablici.

6. Protokol tračanja (eng. Gossipprotocol)

Za komunikaciju unutar čvorova, Cassandra koristi protokol tračanja. Kao što naziv govori, protokol širi informacije kao što se prenosi neka glasina.

Takodje, širenje informacija moze se usporediti sa širenjem virusa.

Dakle, nema centra podataka koja dalje širi informacije, nego se informacija prenosi kroz sve čvorove. Na taj način čvorovi izgradjuju globalnu mapu sastava s malim brojem lokalnih medjudelovanja.
Cassandra koristi protokol tračanja za azuriranje stanja čvorova i njihovih lokacija unutar prstena (klastera).

Proces tračanja pokreće se svake sekunde i izmenjuje informacija s najviše tri čvora unutar klastera. Čvorovi izmjenjuju informacije o sebi i drugim čvorovima o kojima mogu nešto saznati preko drugih sesija tračanja.

Na kraju, svi čvorovi znaju sve o svim čvorovima. Kao sve u Cassandri, svaki trač ima svoj broj verzije, što znači da kad god dva čvora tračaju, starija informacija o čvoru se prepiše se novijom informacijom'

Apache Cassandra postiže strogu konzistentnost, odnosno nije moguce da neki klijent procita staru vriednost nakon što je potvrdeno pisanje nove vriednosti. Time Cassandra garantuje toleranciju na particioniranje i konzistentnost, ali ne više i dostupnost zbog velikog broja cvorova koji je potrebno kontaktirati prilikom citanja i pisanja.

7. Zakljucak

Cassandra koristi arhitekturu peer-to-peer umesto master / slave koja se koristi u RDBMSs.

Ako se glavni cvor zaustavi ili pokvari zbog brojnih zahteva, podređeni cvorovi ostaju beskorisni, dok je u paketu peer-to-peer svaki skup baze podataka jednak i može prihvatiti zahteve bilo kojeg klijenta.

Kao rezultat toga, Cassandra nema niti jednu tačku neuspeha.

8. Literatura

[1] https://repozitorij.etfos.hr/islandora/object/etfos%3A916/datastream/PDF/view

[2] http://www.tfzr.uns.ac.rs/Content/files/0/MASTER%20RAD%20final.pdf

[3] https://www.raf.edu.rs/docs/Diplomski_radovi/Arhitektura_platforme_za_analitiku_podataka_u_oblaku.pdf

[4] http://poincare.matf.bg.ac.rs/~gordana/projektovanjeBP/noSQL_DB.pdf

[5] file:///C:/Users/tstefano/Downloads/1071628.Luka_Kiseljak_Diplomski_Rad_final.pdf