

Cluster rešenja kod MariaDB baze podataka

Studijski program: Računarstvo i informatika

Modul: Softversko inženjerstvo

Predmet: Sistemi za upravljanje bazama podataka

|  |  |
| --- | --- |
| Student: | Profesor: |
| Milica Sokolov,  broj indeksa 1655 | Prof. dr Aleksandar Stanimirović |

Niš, Jun 2024. godina

**Sadržaj:**

[Uvod 2](#_Toc170230003)

[Ključni koncepti MariaDB Galera Clustera 3](#_Toc170230004)

[Pregled Galera Clustera 3](#_Toc170230005)

[Sinhrona replikacija 4](#_Toc170230006)

[Postavljanje MariaDB Galera Clustera pomoću Dockera 6](#_Toc170230007)

[Zahtevi 6](#_Toc170230008)

[Instalacija MariaDB Galera Cluster-a pomoću Docker tehnologije 6](#_Toc170230009)

[Određivanje URL-a čvora 14](#_Toc170230010)

[Prenos stanja čvora (Node provisioning) 15](#_Toc170230011)

[State Snapshot Transfer 15](#_Toc170230012)

[Incremental State Transfer 16](#_Toc170230013)

[Problem split braina 16](#_Toc170230014)

[Arbitrator u Galeri 18](#_Toc170230015)

[Konfiguracija klastera 19](#_Toc170230016)

[Objašnjenje važnih Galera sistemskih promenljivih 19](#_Toc170230017)

[Opšta podešavanja klastera 19](#_Toc170230018)

[Performanse i pouzdanost 20](#_Toc170230019)

[Postavke koje utiču na ponašanje prenosa stanja snimka (SST): 20](#_Toc170230020)

[Rukovanje ograničenjima Galera klastera 21](#_Toc170230021)

[Postavljanje wsrep parametara 21](#_Toc170230022)

[Monitoring i otklanjanje problema 23](#_Toc170230023)

[Skripte za obaveštenja 23](#_Toc170230024)

[Provera statusnih promenljivih 24](#_Toc170230025)

[Zdravlje klastera 24](#_Toc170230026)

[Zdravlje pojedinačnih čvorova 24](#_Toc170230027)

[Zdravlje replikacije 25](#_Toc170230028)

[Performanse mreže 25](#_Toc170230029)

[Balansiranje opterećenja 26](#_Toc170230030)

[Lista ograničenja Galera klastera 26](#_Toc170230031)

[Galera Load Balancer 27](#_Toc170230032)

[Zaključak 30](#_Toc170230033)

[Literatura: 31](#_Toc170230034)

# Uvod

U ovom seminarskom radu bavićemo se temom MariaDB Galera Clustera, tehnologije koja omogućava visokoperformansno i visoko dostupno rešenje za replikaciju podataka koristeći klaster MariaDB čvorova.

Na početku ćemo pružiti osnovne informacije o strukturi i funkcionalnosti Galera Clustera, ističući njegove prednosti i specifičnosti. Zatim ćemo detaljno razmotriti proces instalacije čvorova unutar klastera, koristeći moderne tehnologije kao što je Docker za olakšano upravljanje i postavljanje. Fokusiraćemo se na korake potrebne za inicijalizaciju i konfiguraciju čvorova unutar klastera kako bi se osiguralo njihovo pravilno funkcionisanje. Posebnu pažnju posvetićemo izazovima koji nastaju usled problema "split brain" i kako se isti mogu rešiti upotrebom Galera Arbitratora za održavanje integriteta podataka. Osvrnućemo se i na alate i tehnike za identifikaciju i rešavanje problema koji mogu uticati na performanse klastera. Na kraju ćemo pružiti uputstva za implementaciju i optimizaciju balansiranja opterećenja kako bi se obezbedila efikasnost i pouzdanost sistema.

Cilj ovog rada je da pruži sveobuhvatan pregled MariaDB Galera Clustera, od osnovnog koncepta do naprednih tehnika za upravljanje i optimizaciju performansi.

# Ključni koncepti MariaDB Galera Clustera

Galera Cluster je implementacija klastera za MariaDB i MySQL. MariaDB Galera Cluster je zvanična MariaDB distribucija koja sadrži Galera tehnologiju. Prati iste glavne verzije kao i osnovna MariaDB verzija. Prva verzija je bila 5.5. Druga distribucija, Percona XtraDB Cluster, bazirana je na Percona Serveru.

MariaDB Galera Cluster može biti instaliran i ažuriran iz zvaničnih MariaDB repozitorijuma, ili se generički binarni fajlovi za Linux mogu preuzeti sa MariaDB sajta.

## Pregled Galera Clustera

Galera Cluster je sistem koji omogućava sinhronu replikaciju podataka između servera u klasteru. Svaki čvor u klasteru funkcioniše kao samostalni MariaDB server. To znači da svaki čvor može prihvatiti korisničke zahteve, odgovoriti na upite i lokalno primeniti promene podataka. Nakon što promene budu primenjene na jednom čvoru, one se prenose na sve ostale čvorove u klasteru.

Iako se čvorovi u Galera Clusteru uglavnom ponašaju kao standardni MariaDB serveri, postoje neka ograničenja. Najvažnije je da Galera radi samo na Linuxu. Takođe, samo je InnoDB skladišni motor potpuno podržan, dok MyISAM ima samo eksperimentalnu podršku i njegovo korišćenje se ne preporučuje.

Broj čvorova unutar klastera nije ograničen. Međutim, da bi klaster bio otporan na pad sistema, preporučuje se da ima najmanje tri čvora.

Galera Cluster se ističe visokim nivoom konzistentnosti u poređenju sa većinom MySQL proizvoda koji često koriste labave modele konzistentnosti poput eventualne konzistentnosti. U Galeri, sve promene podataka se primenjuju u strogo definisanom redosledu na svim čvorovima klastera. Ova strogoća redosleda je moguća zbog upotrebe sinhronog modela replikacije.

Ovaj pristup garantuje da Galera Cluster može obezbediti visok nivo tačnosti i doslednosti podataka, što je od suštinskog značaja za aplikacije koje zahtevaju pouzdane operacije sa bazom podataka, poput finansijskih transakcija ili sistema za upravljanje inventarom. Sposobnost da se održi strogi redosled promena podataka na svim čvorovima čini Galera Cluster izuzetno pouzdanim izborom za organizacije koje traže stabilnost i sigurnost u radu sa podacima.

Zbog svoje arhitekture, Galera Cluster se može koristiti za različite svrhe. Na primer, može se koristiti za balansiranje opterećenja jer su svi čvorovi stalno ažurirani. Ako su čvorovi geografski udaljeni, klijenti mogu komunicirati sa najbližim čvorom, što smanjuje kašnjenje. Takođe, može se koristiti za backup jer jedan ili više čvorova uvek mogu biti ažurirani, čime se izbegava gubitak nedavnih podataka. Galera se može koristiti i kao tradicionalni replikacioni sistem gde svi klijenti upituju samo jednog master-a dok su ostali čvorovi pasivni slave-ovi.

Galera Cluster je veoma pogodna za cloud computing zbog svoje funkcije automatskog dodavanja čvorova, što čini skaliranje veoma jednostavnim.

## Sinhrona replikacija

Sinhrona replikacija je metod replikacije podataka koji se koristi u sistemima baza podataka poput MySQL-a i MariaDB-a i osigurava da se sve operacije pisanja (transakcije) sinhronizuju između svih čvorova u klasteru. Za razliku od asinhronih rešenja za replikaciju, poput ugrađene MySQL replikacije, gde ne postoji garancija kada će se promene sa glavnog čvora (master-a) preneti na njegove slavove, sinhrona replikacija zahteva da se sve promene repliciraju u realnom vremenu.

Asinhrona replikacija garantuje da će sve operacije pisanja koje se dese na master čvoru biti prenete na sve slavove, ali ne garantuje kada će se to dogoditi. U praksi, na prometnim mrežama, nije neuobičajeno da neki slavovi zaostaju za masterom sa kašnjenjem od sati ili čak dana.

Sinhrona replikacija osigurava da su master i slavovi uvek sinhronizovani. Transakcije se obrađuju istovremeno i teoretski ne bi trebalo da dolazi do kašnjenja. Kao što je već pomenuto, sinhrona replikacija ima dve ključne posledice: nema gubitka podataka nakon pada master čvora i transakcije se komituju istim redosledom na svim čvorovima.

Implementacija sinhrone replikacije može biti kompleksna i spora kada se koristi tradicionalne metode kao što su distribuirani lock mehanizami ili dvofazni commit. Galera Cluster, međutim, koristi inovativni pristup baziran na sertifikaciji transakcija, koji je efikasniji i brži.

Kada čvor u Galera klasteru primi SQL naredbu, on je odmah izvršava lokalno. Umesto da čeka da svi ostali čvorovi potvrde (commit) ovu operaciju, čvor šalje setove zapisa (promene koje je izvršio) drugim čvorovima u klasteru. Ovi setovi zapisa se prenose paralelno, bez čekanja na potvrdu. Svaki čvor koji primi ove setove zapisa primenjuje ih lokalno, ali ne čini te promene efektivnim odmah. Umesto toga, čvor proverava da li su modifikacije uspešno primenjene i sertifikuje set zapisa. Sertifikacija znači da je čvor uspešno proverio i spreman je da potvrdi (commit) promene. Kada prvi čvor primi potvrdu (commit) od klijenta i svi čvorovi u klasteru sertifikuju set zapisa, te promene postaju efektivne i commit se propagira kroz ceo klaster. Ovaj proces osigurava da su svi čvorovi sinhronizovani bez dugog čekanja na potvrde, što značajno ubrzava rad klastera.

Da bi se smanjila verovatnoća konflikata, transakcije se reorganizuju pre nego što se propagiraju drugim čvorovima. Ovo znači da se setovi zapisa koji se šalju između čvorova optimizuju kako bi se izbegli sukobi. Naredbe koje implicitno izazivaju commit, kao što su određeni SQL upiti, izolovane su od drugih setova zapisa, što dodatno smanjuje rizik od konflikata.

Sinhrona replikacija je omogućena pomoću wsrep-a, API-ja za implementaciju ove funkcije u MySQL-u i MariaDB-u. Wsrep API je skraćenica za Write Set Replication API. Galera se može smatrati implementacijom ovog API-ja, kao i dodatkom za MariaDB i MySQL. Iz tog razloga, Galera se ponekad naziva wsrep provajderom.

Opis wsrep-a na stranici projekta Launchpad kaže:

"Wsrep API definiše skup aplikacionih povratnih poziva i poziva biblioteke replikacije neophodnih za implementaciju sinhronog replikacije setova zapisa transakcionih baza podataka i sličnih aplikacija. Cilj mu je da apstrahuje i izoluje implementaciju replikacije od detalja aplikacije. Iako je glavna meta ovog interfejsa certifikaciona višemasterska replikacija, podjednako je pogodan za asinhronu i sinhronu replikaciju master/slave."

# Postavljanje MariaDB Galera Clustera pomoću Dockera

Ovaj odeljak opisuje kako postaviti MariaDB Galera Clustera pomoću Dockera. Međutim, prvo ćemo razmotriti zahteve klastera.

## Zahtevi

Kao što je ranije pomenuto, svi čvorovi klastera moraju raditi na Linux sistemima. Nema posebnih zahteva za hardverom. Tokom normalnog rada, dodatna količina memorije koju Galera replikacija koristi je minimalna i ne bi trebalo da pravi veliku razliku. Jedina izuzetak je kada se celokupna baza podataka kopira sa jednog čvora na drugi, što troši puno memorije. Ovo se dešava kada novi čvor pristupa klasteru ili kada se odvojeni čvor ponovo poveže sa klasterom. Međutim, pri izboru hardvera za pojedinačne čvorove, važno je imati na umu da će klaster biti toliko brz koliko je njegov najsporiji čvor.

Veza između čvorova mora biti dovoljno brza za opterećenje koje planiramo pokrenuti. Ako je moguće, MariaDB Galera klaster treba pokrenuti na podmreži koja ne sadrži druge domaće vhosts. Pošto Galera replikacija podrazumeva stalnu komunikaciju između svih čvorova, prisustvo nepovezanih mrežnih saobraćaja na istoj mreži će smanjiti performanse. Takođe, sprečavanje spoljnih domaćina da direktno povežu sa čvorovima klastera je važno poboljšanje sigurnosti.

Naravno, čvorovi koji prihvataju konekcije od klijenata će zahtevati više memorije za izvršavanje upita. Moguće je takođe koristiti balanser opterećenja za optimalnu distribuciju upita preko više čvorova. Ova tehnika će biti detaljno opisana kasnije u ovom radu.

## Instalacija MariaDB Galera Cluster-a pomoću Docker tehnologije

Pre nego što započnemo sa instalacijom Galera Clustera, potrebno je da instaliramo Docker na naš sistem. Docker je kontejnerska platforma koja omogućava jednostavno upravljanje aplikacijama u izolovanim okruženjima.

1. **Preuzimanje Docker slike za MariaDB Galera Cluster**

MariaDB Galera Cluster ima zvaničnu Docker sliku koja se može jednostavno preuzeti sa Docker Hub-a. Ova slika već sadrži sve potrebne komponente za Galera Cluster, uključujući i Galera wsrep provajder.

docker pull bitnami/mariadb-galera:latest

Ova komanda će preuzeti poslednju verziju MariaDB Galera Cluster Docker slike sa Docker Hub-a.

1. **Bootstrap Klastera**

Ovaj korak podrazumeva pokretanje prvog (bootstrap) čvora klastera. Ovaj čvor će inicijalizovati klaster i omogućiti ostalim čvorovima da mu se pridruže. Pokretanje bootstrap čvora je ključno jer on kreira inicijalnu konfiguraciju klastera.

Komanda za Pokretanje Bootstrap Čvora:

docker run --name mariadb-galera-0 ^

-v "C:\1. Moje datoteke\1. Master\Sistemi za upravljanje bazama podataka - Seminarski rad I\galera:/var/lib/mysql" ^

-e MARIADB\_GALERA\_CLUSTER\_BOOTSTRAP=yes ^

-e MARIADB\_GALERA\_CLUSTER\_NAME=my\_galera ^

-e MARIADB\_GALERA\_MARIABACKUP\_USER=my\_mariabackup\_user ^

-e MARIADB\_GALERA\_MARIABACKUP\_PASSWORD=my\_mariabackup\_password ^

-e MARIADB\_ROOT\_PASSWORD=my\_root\_password ^

-e MARIADB\_USER=my\_user ^

-e MARIADB\_PASSWORD=my\_password ^

-e MARIADB\_DATABASE=my\_database ^

bitnami/mariadb-galera:latest

Objašnjenje:

* --name mariadb-galera-0:

Postavlja ime kontejnera na mariadb-galera-0. Ovo ime je važno za identifikaciju kontejnera unutar Docker okruženja i za povezivanje drugih čvorova sa ovim inicijalnim čvorom.

* -e MARIADB\_GALERA\_CLUSTER\_BOOTSTRAP=yes:

Označava da se ovaj čvor koristi za inicijalizaciju klastera. Ovo omogućava da ovaj čvor kreira novu Galera klaster instancu.

* -e MARIADB\_GALERA\_CLUSTER\_NAME=my\_galera:

Definiše ime klastera na my\_galera. Svi čvorovi koji se pridružuju ovom klasteru moraju koristiti isto ime kako bi se povezali sa ovim klasterom.

* -e MARIADB\_GALERA\_MARIABACKUP\_USER=my\_mariabackup\_user i

-e MARIADB\_GALERA\_MARIABACKUP\_PASSWORD=my\_mariabackup\_password:

Postavlja korisnika (my\_mariabackup\_user) i lozinku (my\_mariabackup\_password) za MariaDB Backup. Ovi podaci se koriste za sigurnosne kopije i replikaciju podataka unutar klastera.

* -e MARIADB\_ROOT\_PASSWORD=my\_root\_password:

Postavlja lozinku za root korisnika na my\_root\_password. Ova lozinka je potrebna za administrativne zadatke i pristup svim bazama podataka na ovoj instanci MariaDB.

* -e MARIADB\_USER=my\_user,

-e MARIADB\_PASSWORD=my\_password,

-e MARIADB\_DATABASE=my\_database:

Ovi parametri postavljaju običnog korisnika (my\_user), lozinku za tog korisnika (my\_password), i kreiraju bazu podataka (my\_database). Ovi podaci su korisni za aplikacije koje će koristiti ovu bazu podataka za svoje potrebe.

* bitnami/mariadb-galera:latest:

Ovo je Docker slika koja se koristi za pokretanje MariaDB Galera klastera. bitnami/mariadb-galera:latest označava da će se koristiti najnovija verzija slike koju pruža Bitnami, koja uključuje sve potrebne komponente za pokretanje MariaDB Galera klastera.

1. **Dodavanje čvorova u klaster**

Sada ćemo dodati nove čvorove u klaster. Svaki novi čvor će se pridružiti klasteru koji je inicijalizovan u prethodnom koraku.

docker run --name mariadb-galera-1 --link mariadb-galera-0:mariadb-galera \

-e MARIADB\_GALERA\_CLUSTER\_NAME=my\_galera ^

-e MARIADB\_GALERA\_CLUSTER\_ADDRESS=gcomm://mariadb-galera ^

-e MARIADB\_GALERA\_MARIABACKUP\_USER=my\_mariabackup\_user ^

-e MARIADB\_GALERA\_MARIABACKUP\_PASSWORD=my\_mariabackup\_password ^

-v "C:/1. Moje datoteke/1. Master/Sistemi za upravljanje bazama podataka - Seminarski rad I/galera1:/var/lib/mysql" ^

bitnami/mariadb-galera:latest

Objašnjenje:

* --name mariadb-galera-1: Postavlja ime novog čvora.
* --link mariadb-galera-0:mariadb-galera: Povezuje novi čvor sa prvim čvorom.
* -e MARIADB\_GALERA\_CLUSTER\_NAME=my\_galera: Definiše ime klastera.
* -e MARIADB\_GALERA\_CLUSTER\_ADDRESS=gcomm://mariadb-galera: Adresa klastera za povezivanje.
* Ostali parametri ostaju isti kao i u prvom koraku.

Možemo dodati više čvorova na sličan način promenom imena kontejnera (--name).

1. **Provera statusa klastera**

Za provere i osnovne operacije nad klasterom, možemo pristupiti kontejnerima i koristiti MariaDB CLI.

docker exec -it mariadb-galera-0 /bin/bash

docker exec -it mariadb-galera-1 /bin/bash

Unutar kontejnera:

ip a - Prikazuje IP adrese kontejnera

ps -ef | grep mysql - Prikazuje MySQL procese

mysql -u my\_user -p -h 172.17.0.3 - Pristupa Maria DB instanci

1. **Povezivanje s MariaDB serverom unutar Docker kontejnera**

docker exec -it mariadb-galera-0 mysql -u root -p

Ova komanda omogućava interaktivno povezivanje sa MySQL ili MariaDB serverom koji se pokreće unutar Docker kontejnera mariadb-galera-0. Ovom komandom otvarate interaktivnu sesiju sa MySQL/MariaDB serverom, gde možete izvršavati SQL upite i administrativne zadatke. Navođenjem opcije -u root specificirate da se povezujete kao korisnik root, a opcija -p traži unos lozinke nakon pokretanja komande radi sigurnosti pristupa serveru.

1. **Provera statusa klastera**

Nakon što je čvor podešen, možda ćemo želeti da proverimo da li ispravno radi. Galera pruža neke statusne varijable za dijagnostičke svrhe. Sve Galera-povezane statusne varijable imaju nazive koji počinju sa wsrep\_, kao što je prikazano u nastavku:

SHOW STATUS LIKE 'wsrep%';

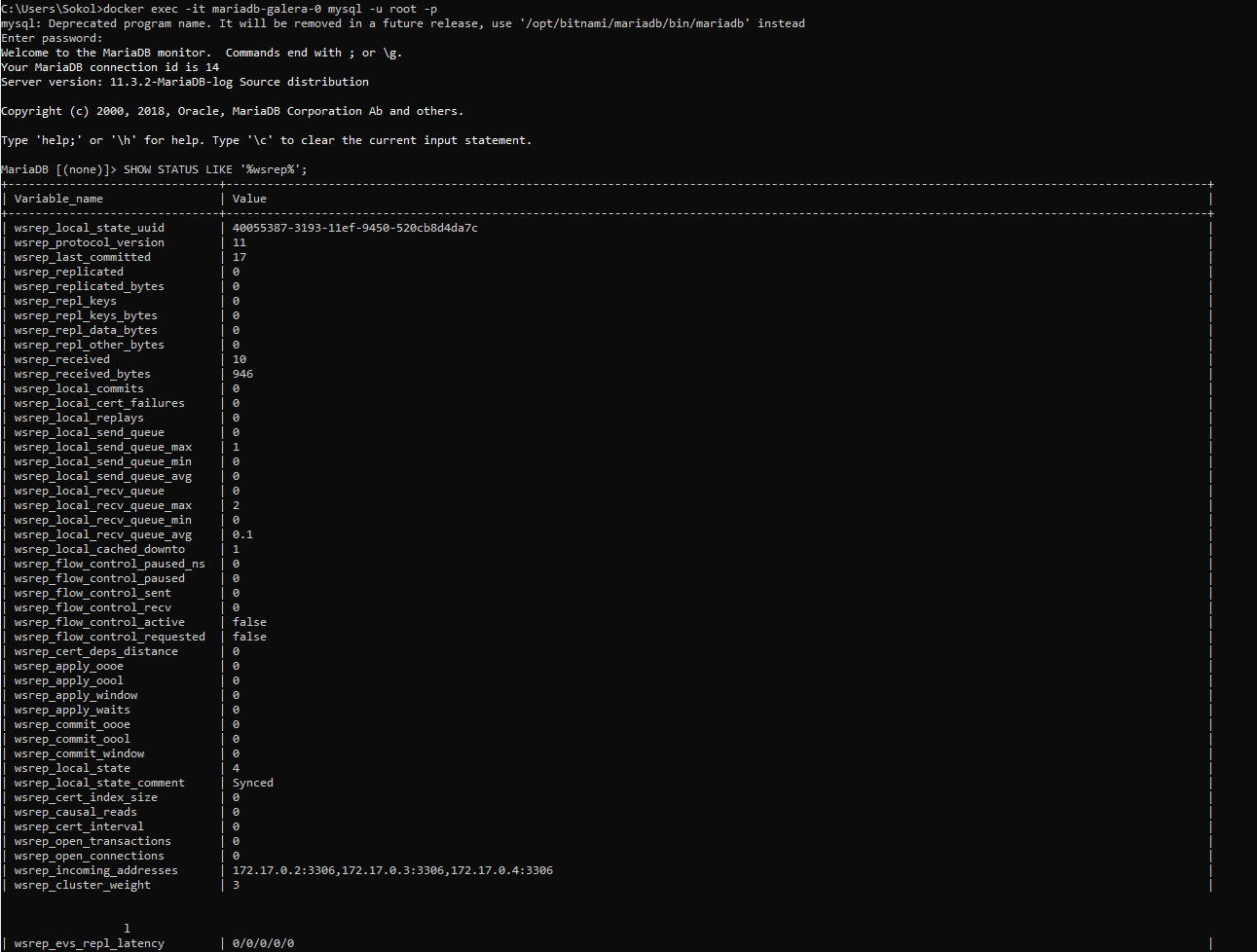
Proverite sledeće varijable:

* wsrep\_ready: Pokazuje da li je čvor povezan sa klasterom i spreman da primi replikacione događaje (ON je željena vrednost). Jedina druga vrednost može biti OFF.
* wsrep\_connected: Pokazuje da li je čvor povezan sa wsrep provajderom.
* wsrep\_cluster\_status: Pokazuje da li je čvor povezan sa klasterom. Ako nijedan drugi čvor nije povezan, vrednost je Disconnected. Inače, videćemo Primary ili Non Primary.
* wsrep\_cluster\_size: Pokazuje broj čvorova u klasteru.

Takođe, možemo proveriti i koje sve baze podataka su dostupne slewdecom naredbom:

SHOW DATABASES; # Prikazuje baze podataka

U primeru na slici 1. vidimo da smo se povezali sa mariadb-galera-0 cvorom, I zatim izvrsili naredbu za prikaz statusa. Iz rezultata komande SHOW STATUS LIKE '%wsrep%' za Galera klaster možemo videti trenutno stanje klastera: UUID-ove koji identifikuju stanje čvora i klastera, verziju Galera protokola (u ovom slučaju 11), status klastera (primarni), broj čvorova (3), stanje povezanosti (ON), kao i različite metrike vezane za replikaciju i performanse. Ovi podaci su ključni za praćenje zdravlja klastera, identifikaciju problema i održavanje stabilnosti i performansi Galera baze podataka.



Slika 1. Informacije o stanju Galera klastera

1. **Kreiranje tabele i unos podataka**

Nakon što izaberemo odgovarajuću bazu podataka u jednom od čvorova Galera klastera, možemo kreirati tabelu example sa definicijom kolona id kao AUTO\_INCREMENT i name kao VARCHAR. Nakon toga, dodajemo uzorak podataka u tu tabelu.

CREATE TABLE example (

id SMALLINT UNSIGNED NOT NULL AUTO\_INCREMENT,

name VARCHAR(20) NOT NULL,

CONSTRAINT pk\_example PRIMARY KEY (id)

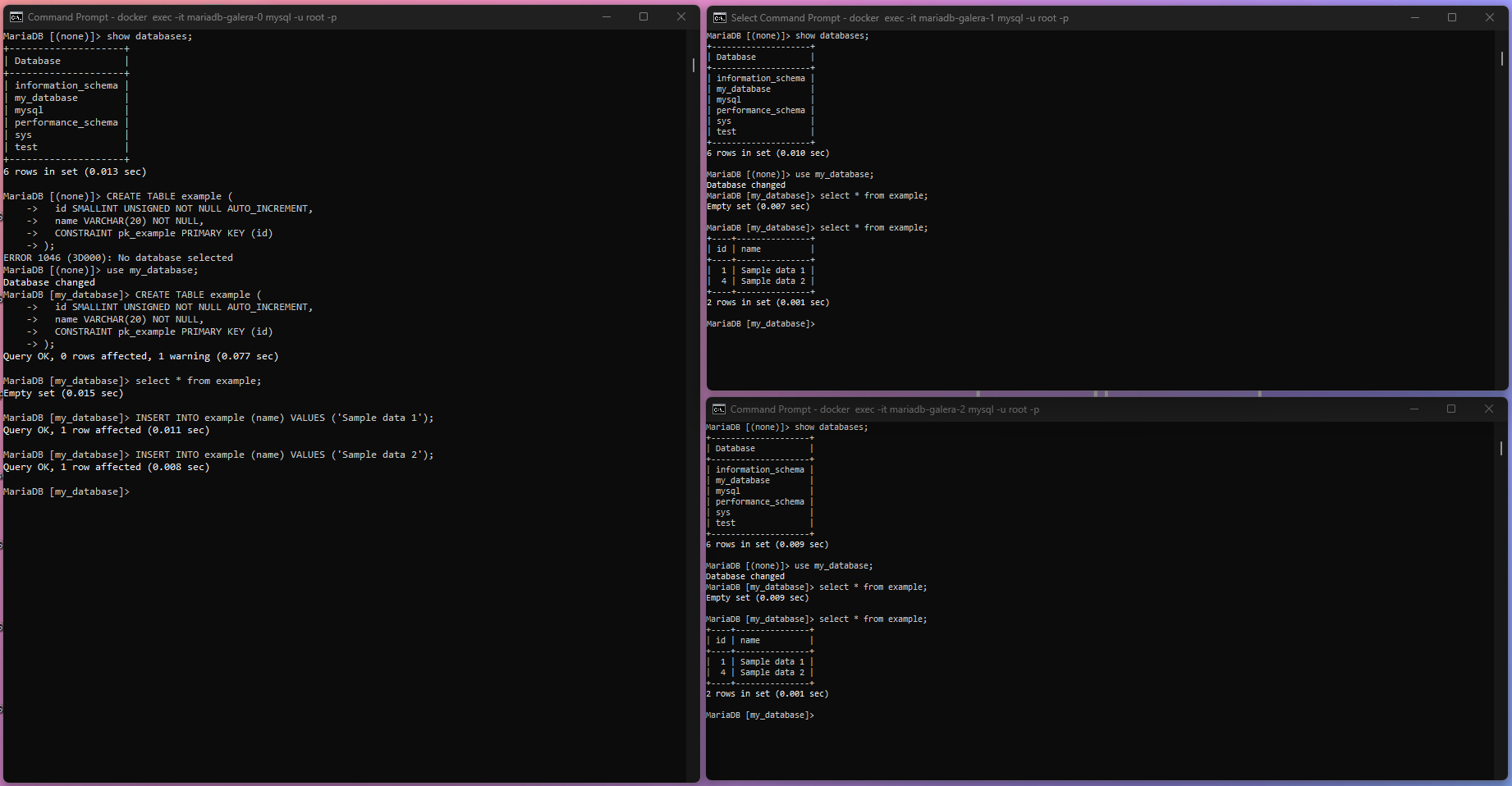
);

INSERT INTO example (name) VALUES ('Sample data 1');

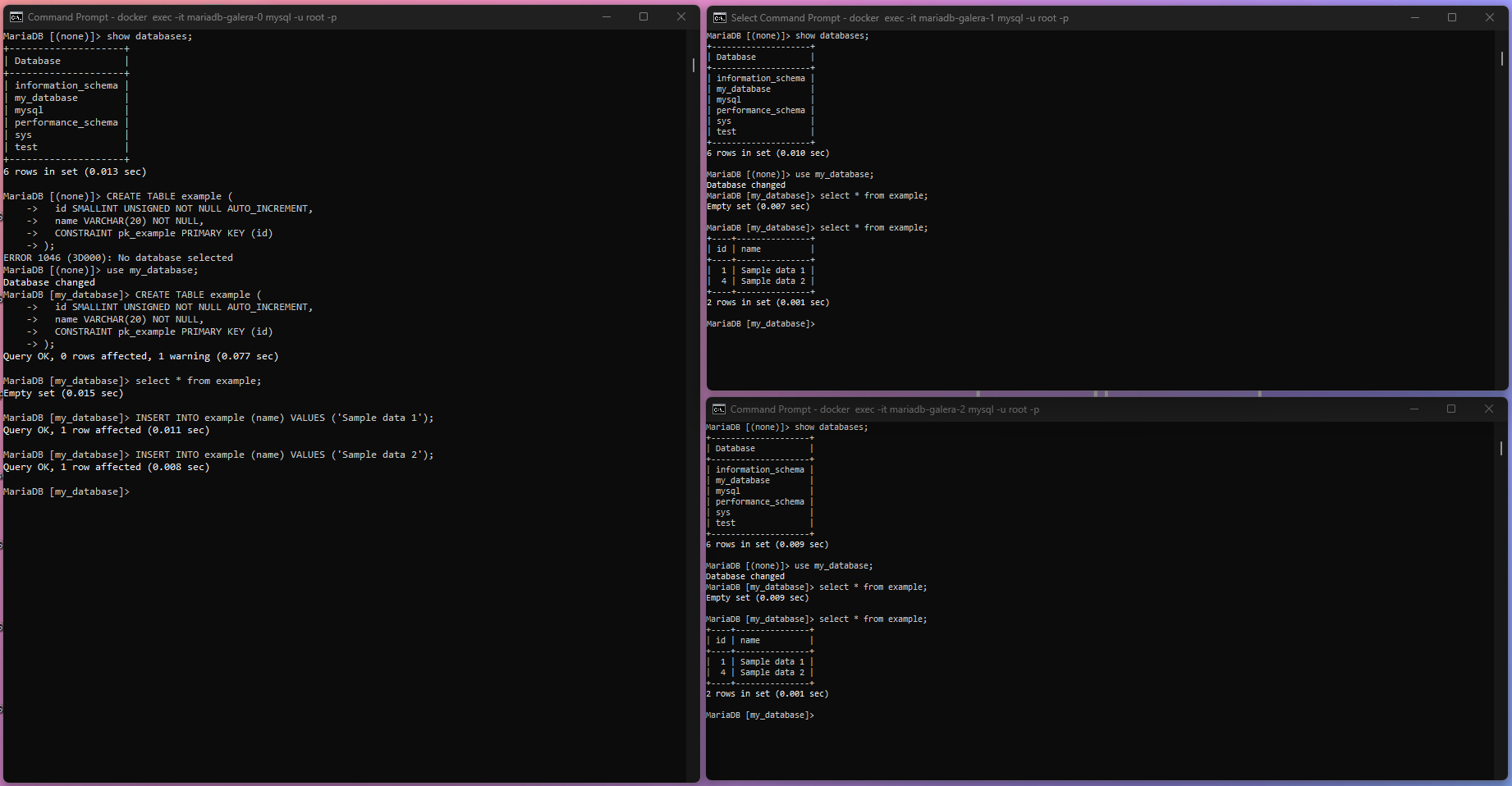
INSERT INTO example (name) VALUES ('Sample data 2');

Ako istu bazu izaberemo i u drugim čvorovima, možemo izvršiti SELECT upit kako bismo videli sadržaj tabele example koji je kreiran u prvom čvoru. Ovo nam omogućava da proverimo da li su podaci pravilno replikovani između čvorova klastera, osiguravajući konzistentnost i dostupnost podataka u celom Galera klasteru.

Na slici 2 i 3 možemo videti da smo u MySQL/MariaDB kontejneru mariadb-galera-0 odabrali bazu podataka my\_database. U toj bazi smo izvršili SQL komande za kreiranje tabele example sa poljima id kao AUTO\_INCREMENT i name kao VARCHAR. Takođe smo uneli uzorke podataka u tu tabelu koristeći INSERT naredbe. Nakon toga, koristeći istu bazu i u drugim čvorovima Galera klastera, mogli smo izvršiti SELECT upit kako bismo pristupili podacima koji su uneti u tabelu example u prvom čvoru. Ovo ilustruje efikasnost replikacije podataka između čvorova klastera, što omogućava konsistentnost i dostupnost podataka u celom Galera klasteru.

****

Slika 2. Kreiranje tabele i dodavanje podataka na cvoru mariadb-galera-0

****

Slika 3. Prikaz tabele nakon dodavanja podataka u MariaDB Galera klasteru na cvoru mariadb-galera-1

1. **Podešavanje dozvola**

Nakon što se čvor pokrene, potrebno je podesiti određene dozvole. Svaki čvor u klasteru mora dozvoliti da se drugi čvorovi mogu povezati kao root kako bi kreirali kopije baza podataka po potrebi. Ovaj mehanizam se naziva node provisioning, o čemu će biti reči u delu Node provisioning kasnije u ovom poglavlju. Dakle, izvršićemo sledeće naredbe na svakom čvoru:

GRANT ALL PRIVILEGES ON **.** TO 'username'@'localhost' IDENTIFIED BY 'password'; SELECT user FROM mysql.user GROUP BY user;

Ovim postupkom smo dokazali da je Galera klaster omogućio efikasno deljenje podataka između više čvorova, omogućavajući nam da kreiramo i pristupamo istim podacima u različitim čvorovima klastera. Time se potvrđuje pouzdanost i sinhronizacija podataka unutar klastera, što je ključno za visoku dostupnost i skalabilnost baza podataka.

## Određivanje URL-a čvora

Kao što je prethodno objašnjeno, za pokretanje novog čvora ili ponovno pokretanje čvora nakon pada, potrebno je specificirati URL drugog čvora. Dakle, administrator baza podataka (DBA) treba da zna kako da odredi URL čvora.

Formalna sintaksa Galera URL-a je sledeća:

<schema>://<address><:port>[?option=value[&option=value …]]

Podržana su dva šema:

* gcomm: Ovo je šema koja se koristi za potpuno funkcionalan Galera klaster. Uvek se mora koristiti u produkciji.
* dummy: Ovo je šema koja se koristi za testiranje Galera konfiguracije. Ako se koristi, podaci se ne replikiraju.

Adresa može biti IP adresa ili naziv hosta, opciono praćena brojem porta. Podrazumevani port je 4567, na primer, 214.140.10.5:9999.

Moguće je navesti više adresa razdvojenih zarezom, na primer, 214.140.10.5,214.140.10.6. Takođe je moguće koristiti multicast adrese, kao što su IPv4 ili IPv6 adrese čiji je poslednji deo 1, kako bi se identifikovali svi hostovi u podmreži.

Skup opcija može biti naveden, razdvojen tačko-zarezom. Ove opcije su iste kao one koje su sadržane u wsrep\_provider\_options, što će biti opisano kasnije. Da bi ovi parametri bili primenjeni nakon restartovanja čvora, moraju biti navedeni u konfiguracionoj datoteci, a ne u URL-u.

Neki primeri URL-ova su sledeći:

gcomm://server\_name

gcomm://214.140.10.5,214.140.10.6

gcomm://server\_name:9999,214.140.10.5,214.140.10.6

## Prenos stanja čvora (Node provisioning)

Prenosa stanja čvora sastoji se od kopiranja potpunog rezervnog snimka podataka sa jednog čvora na drugi. Rezervni snimak se često naziva snapshot ili stanje, kako bi se istaklo da je u pitanju konzistentna verzija podataka u tačno određenom trenutku. Čvor koji šalje svoje stanje naziva se donor, dok se čvor koji ga prima naziva joiner, jer se ova operacija dešava kada čvor pristupa klasteru. Ovo se može desiti jer je nova mašina dodata ili jer je postojeći čvor pao i, nakon ponovnog pokretanja, treba da primi najnovije promene podataka.

Postoje dva glavna metoda provizioniranja čvora:

* **Transfer stanja snimka (State Snapshot Transfer - SST)** sastoji se od slanja potpunog snimka
* **Inkrementalni transfer stanja (Incremental State Transfer - IST)** sastoji se od slanja modifikacija

### State Snapshot Transfer

Metod prenosa stanja čvora koji se koristi kada novi čvor pristupa klasteru jer ne sadrži podatke. Postoje dva načina izvršenja SST-a:

1. **mysqldump**: Ovaj metod koristi alat mysqldump za generisanje SQL naredbi potrebnih za ponovno kreiranje baze podataka na drugom čvoru. Ovaj metod je sporiji jer obično zahteva veliku količinu mrežnog saobraćaja. Donor čvor se čini samo za čitanje putem globalnog zaključavanja tokom celog trajanja transfera stanja. Takođe, ovaj metod zahteva da je joiner čvor već pokrenut. Upotreba mysqldump-a je neophodna ako čvorovi koriste različite verzije MariaDB ili različite rasporede direktorijuma za podatke.
2. **rsync**, **rsync\_wan**, i **xtrabackup**: Ovi alati se koriste za kopiranje datoteka podataka sa donor čvora na joiner čvor. Ovaj metod je znatno brži. Datoteke se mogu kopirati korišćenjem rsync-a, koji kopira samo datoteke koje su izmenjene; rsync\_wan metod koristi rsync sa algoritmom delta transfera, koji se koristi za kopiranje podataka preko Wide Area Network (WAN) ili sporih mreža, ali je sporiji u drugim situacijama. Percona XtraBackup pravi kopiju tabela bez zaključavanja servera. Metod rsync je brži od XtraBackup-a ali je blokirajući metod. Ovi metodi zahtevaju da postavke koje utiču na način čuvanja datoteka, kao što su @@innodb\_file\_per\_table ili @@innodb\_file\_format, imaju iste vrednosti na oba čvora. Bitno je napomenuti da ako se koristi jedan od ovih metoda, joiner čvor ne sme biti inicijalizovan pre transfera.

Da bi se izabrao metod koji će se koristiti za SST, opcija wsrep\_sst\_method može se postaviti u konfiguracionoj datoteci joiner čvora.

### Incremental State Transfer

Svi skupovi zapisa koje je potvrdio čvor upisuju se u poseban keš nazvan Galera keš (GCache). Njegova svrha je ubrzanje operacija ulazno/izlaznih podataka. Kada čvor izgubi vezu i ponovno se pokrene, moguće je da su skupovi zapisa izvršeni od strane drugih čvorova potpuno smešteni u GCache-u barem jednog čvora. U ovom slučaju, koristi se metoda Inkrementalnog Transfera Stanja (Incremental State Transfer) kako bi se novi čvor ažurirao. Ova metoda ima dve važne prednosti. Prenos stanja podataka je mnogo brže u odnosu na SST jer se joineru šalju samo nedavno izmenjeni podaci. Nema potrebe za zaključavanjem donora, što mu omogućava da nastavi sa replikacijom događaja koje prima tokom transfera stanja.

## Problem split braina

Klasteri svake vrste moraju biti spremni da reše problem koji se naziva split brain. Da bismo razumeli ovaj problem i mogući stvarni slučaj, zamislimo klaster baza podataka koji je raspoređen u dva data centra. Pretpostavimo da jedan od data centara izgubi internet konekciju. Klaster se sada deli na dve particije. Ipak, čvorovi i dalje rade, lokalni klijenti mogu da se povežu i šalju upite. Ako klaster nije spreman da se nosi sa ovakvom situacijom, oba data centra će verovatno nastaviti da modifikuju iste skupove podataka. Kada se internet konekcija ponovo uspostavi za diskonektovani data centar, pojaviće se mnogo konflikata u bazi podataka. Klaster može automatski da reši ove konflikte; ovo se naziva optimistički pristup problemu split brain. Ako klaster nije u stanju da reši konflikte, možemo reći da se desila katastrofa.

Galera koristi pesimistički pristup problemu split brain. Tehnika koju koristi naziva se kvorum sa težinama i varijacija je algoritma kvoruma-saglasnosti. Pogledajmo kako ova tehnika funkcioniše.

Svi čvorovi u klasteru čuvaju broj čvorova u klasteru. Ako se dodaje novi čvor, veličina klastera se povećava. Ako se čvor elegantno isključi, on komunicira sa drugim čvorovima da napušta klaster. Međutim, ako čvor prestane sa radom ili dođe do trajnog problema sa mrežom, on ne može ništa da komunicira; jednostavno postaje nedostupan. Ako je čvor nedostupan određeno vreme, drugi čvorovi pretpostavljaju da više nije dostupan. Podrazumevano vreme čekanja je 5 sekundi, ali se može prilagoditi putem opcije evs.suspect\_timeout u varijabli wsrep\_provider\_options servera.

Kada neki čvorovi otkriju da drugi čvor nije dostupan, koristi se algoritam kvoruma. Ako je više od polovine čvorova u klasteru i dalje dostupno, trenutna particija klastera i dalje je primarna. Ovo je podrazumevana situacija čak i pre nego što se desi bilo koji pad. To znači da trenutna particija može nastaviti sa svim normalnim operacijama.

Ako je dostupno samo polovina ili manje od polovine originalnog broja čvorova, trenutna particija postaje nepriamrna. Čvorovi i dalje mogu prihvatiti konekcije i izvršavati upite koje šalju klijenti, ali njihove baze podataka postaju samo za čitanje. Kao što je ranije pomenuto, možemo proveriti da li je trenutna particija klastera primarna ili ne koristeći statusnu varijablu @@wsrep\_cluster\_status.

Ako klaster ima samo dva čvora i izgubi se konekcija između njih, nijedna particija neće biti veća od druge. Isti problem se javlja i ako jedan čvor prestane sa radom. U oba slučaja, nijedna od rezultirajućih particija neće biti primarna, pa ni jedan čvor neće moći da upisuje podatke. Ovo objašnjava zašto svi Galera klasteri treba da se sastoje od najmanje tri čvora.

Ovaj algoritam garantuje da, ako se klaster podeli na dve ili više particija, samo jedna od njih će biti primarna. Ni u jednoj situaciji više od jedne particije ne može da modifikuje podatke.

Ipak, već smo pomenuli da Galera koristi varijaciju algoritma kvoruma nazvanu kvorum sa težinama. To znači da čvorovima mogu biti dodeljene različite težine. Kada je čvor nedostupan, Galera zapravo ne računa veličinu trenutne particije; umesto toga, računa težinu particije, koja je zbir težina svih pojedinačnih čvorova. Podrazumevano, težina svakog čvora je 1, pa je težina particije identična njenoj veličini.

Različite težine mogu se dodeliti pomoću opcije pc.weight. Dozvoljeni opseg je od 0 do 255. Ako je težina čvora 0, onda čvor ne utiče na rezultat formule kvoruma. Ako se poveća težina grupe čvorova, a ti čvorovi izgube internet konekciju, grupa će verovatno postati primarna. Ovo ima smisla, na primer, ako jedan data centar ima manje servera baza podataka od drugih, ali je važno da nastavi sa modifikacijom svojih podataka.

Čak i ako su težine čvorova eksplicitno podešene, uvek je garantovano da neće postojati više od jedne particije koja će postati primarna.

Algoritam kvoruma sa težinama može se onemogućiti omogućavanjem opcije pc.ignore\_quorum u varijabli wsrep\_provider\_options servera. Međutim, imajte na umu da ako je opcija pc.ignore\_quorum omogućena, može doći do problema split braina. U tom slučaju, potrebno je znati kako rešiti konflikte bez pomoći Galere; na primer, u nekim slučajevima može biti prihvatljivo prepisati promene izvršene u particiji.

## Arbitrator u Galeri

Arbitrator je poseban tip čvora dizajniran da pomogne u rešavanju problema split braina.

Komunicira sa ostatkom klastera kao da je normalan čvor, ali ne replicira nikakve podatke. Njegova jedina svrha je da poveća veličinu particije sa kojom može komunicirati, čime je potencijalno čini primarnom particijom.

Na primer, pretpostavimo da imamo samo dva čvora. Kao što je objašnjeno ranije, ako jedan od njih prestane sa radom ili ako se veza izgubi između njih, rezultirajuće particije neće biti primarne. Međutim, ako imamo arbitra, čvor koji nije prestao sa radom ili izgubio vezu moći će da komunicira sa arbitrom. Tehnički, čvorovi će formirati klaster od dva čvora, koji će biti primarna particija.

Složeniji primer je kada dva data centra formiraju Galera klaster i svaki data centar sadrži isti broj čvorova. Ovaj primer, iako uključuje veći broj servera, skoro je identičan prethodnom; ako se izgubi veza između data centara, ni jedan od njih neće moći postati primarna particija. Problem se može rešiti bez dodavanja novog čvora. Možemo postaviti arbitra, koji se ne nalazi ni u jednom od ta dva data centra. Ako jedan od njih izgubi internet konekciju, drugi će i dalje komunicirati sa arbitrom i postati primarna particija.

Za pokretanje arbitra, potrebno je pozvati binarnu datoteku Galera Arbitrator Daemon (garbd). Njegove sistemne promenljive i wsrep parametri su isti kao opcije za obične čvorove. Izuzetak ovde je što su wsrep parametri u grupi repl izostavljeni kod arbitra, jer nemaju smisla za čvor koji ne replicira ništa.

Da bismo koristili konfiguracionu datoteku sa garbd, možemo ga pokrenuti na sledeći način:

garbd --cfg /putanja/do/garb.cnf --daemon

Da bismo zaustavili arbitra, možemo zaustaviti proces garbd. Ovo je bezbedno, pošto arbitar ne piše nikakve podatke.

# Konfiguracija klastera

Čvorovi MariaDB Galera klastera mogu se konfigurisati postavljanjem opštih MariaDB sistemskih promenljivih i Galera-specifičnih promenljivih. Najvažnija promenljiva je wsrep\_provider\_options; može se koristiti za postavljanje mnogih wsrep parametara, razdvojenih tačka-zarezom. Ova promenljiva je dinamička.

Najvažnije promenljive i parametri su objašnjeni u ovom odeljku.

## Objašnjenje važnih Galera sistemskih promenljivih

Sve Galera-specifične promenljive imaju sufiks 'wsrep\_'. Da bismo ih izlistali, možemo koristiti sledeći upit:

SHOW VARIABLES LIKE 'wsrep%';

Sledi diskusija o najvažnijim Galera sistemskim promenljivim. One su grupisane kako bi se diskusija učinila jasnijom.

### Opšta podešavanja klastera

Sledeća lista objašnjava značenje opštih podešavanja klastera:

* wsrep\_provider: Ovo je putanja do wsrep biblioteke.
* wsrep\_cluster\_address: Ovo je adresa jednog ili više čvorova klastera. Kako je već objašnjeno, postavljanje ove vrednosti je neophodno za dodavanje čvora u klaster. Ova promenljiva nije dinamička.
* wsrep\_cluster\_name: Ovo je ime klastera. Čvorovi odbijaju da se povežu sa drugim čvorovima ako ne pripadaju istom klasteru. Ova promenljiva je dinamička i može se koristiti za deljenje klastera.
* wsrep\_node\_address: Ovo može biti korišćeno da se eksplicitno postavi adresa trenutnog čvora. Ova promenljiva nije dinamička.
* wsrep\_node\_name: Ovo postavlja ime čvora. Podrazumevano se koristi ime hosta.
* wsrep\_on: Ovo određuje da li čvor replikuje podatke. Može se koristiti da privremeno zaustavi rad čvora.

### Performanse i pouzdanost

Sledeća lista prikazuje podešavanja koja se mogu koristiti za podešavanje performansi i pouzdanosti:

* wsrep\_data\_home\_dir: Ovo postavlja direktorijum podataka za trenutni čvor.
* wsrep\_slave\_threads: Ovo određuje broj istovremenih niti koje se koriste za replikaciju. Ovo može ili ne mora ubrzati većinu operacija; posebno je korisno za ubrzanje sinhronizacije čvorova. Preporučena minimalna vrednost je 4 puta broj jezgara CPU-a, ali optimalna vrednost ponekad može biti znatno veća. Ova promenljiva nije dinamička, tako da može biti potrebno više puta restartovati čvor da bi se testirala performansa sa različitim vrednostima. Imajte na umu da ako je ova vrednost veća od 1, innodb\_locks\_unsafe\_for\_binlog mora biti postavljen na 1.
* wsrep\_causal\_reads: Ako je ova promenljiva postavljena na OFF, uzrokuje da brži čvorovi primene skup za pisanje i započnu izvršenje novih instrukcija bez čekanja sporijih čvorova. Ovo je podrazumevana vrednost, ali može uzrokovati male neusaglašenosti u kratkom periodu. Postavljanje ove promenljive na ON sprečava ove neusaglašenosti, ali povećava latenciju. Podrazumevana vrednost je OFF.
* wsrep\_max\_ws\_size: Ovo određuje maksimalnu veličinu skupa za pisanje, izraženu u bajtovima. Ova promenljiva nije dinamička.
* wsrep\_max\_ws\_rows: Ovo postavlja maksimalan broj redova u jednom skupu za pisanje. Ova promenljiva nije dinamička.
* wsrep\_retry\_autocommit: Ovo pomaže u rešavanju čestih sukoba unutar klastera. Postavlja maksimalan broj pokušaja kada transakcija ne uspe zbog takvih sukoba. Ova promenljiva nije dinamička.
* wsrep\_load\_data\_splitting: Ovo deli dugačke LOAD DATA INFILE instrukcije na više transakcija, čineći ih bržim ali manje pouzdanim.

### Postavke koje utiču na ponašanje prenosa stanja snimka (SST):

* wsrep\_sst\_donor: Ovo je lista čvorova odvojenih zarezom koji mogu biti korišćeni kao donor. Ovde se koriste imena čvorova umesto njihovih adresa.
* wsrep\_sst\_donor\_rejects\_queries: Ovo određuje da li trenutni čvor odbija da deluje kao donor za druge čvorove. Podrazumevana vrednost je OFF, što znači da može delovati kao donor.
* wsrep\_sst\_method: Ovo određuje metod koji će se koristiti za SST, kako je objašnjeno u delu o Prenosu stanja snimka (SST).
* wsrep\_sst\_auth: Ovo je relevantno samo za SST metode koje podrazumevaju vezu sa pokrenutim serverom, kao što su xtrabackup ili mysqldump. Ova promenljiva sadrži korisničko ime i lozinku za autentifikaciju, razdvojene dvotačkom, na primer, root

### Rukovanje ograničenjima Galera klastera

Sledeće postavke su korisne kada se suočavate sa ograničenjima Galera klastera:

* wsrep\_replicate\_myisam: Ovo određuje da li će se koristiti eksperimentalna replikacija MyISAM tabela. Podrazumevana vrednost je OFF, što znači da će MyISAM tabele biti jednostavno ignorisane.
* wsrep\_convert\_LOCK\_to\_trx: Ovo ponekad može pomoći pri prelasku aplikacija sa MyISAM na InnoDB. Ako je podešeno na ON, izjave LOCK TABLES i UNLOCK TABLES se tiho konvertuju u START TRANSACTION i COMMIT. Podrazumevana vrednost je OFF. Ako omogućimo ovu funkciju, trebalo bi da proverimo da li naše aplikacije ispravno rade.
* wsrep\_certify\_nonPK: Ovo instruiše Galeru da automatski dodaje primarne ključeve za tabele koje ih nemaju. Ovo rešava nekoliko problema sa replikacijom. Podrazumevana vrednost je ON.

## Postavljanje wsrep parametara

Mnoge wsrep postavke mogu se specificirati korišćenjem promenljive wsrep\_provider\_options. Postavke se moraju razdvojiti sa tačka-zarezom, kao što je prikazano u sledećoj sintaksi:

SET wsrep\_provider\_options = 'opcija=vrednost;opcija=vrednost';

Slično kao sa server promenljivama, wsrep opcije mogu biti dinamične ili ne. Ako opcija nije dinamična, čvor se mora ponovo pokrenuti kako bi se promenila njena vrednost.

Skoro sve nazive opcija karakteriše prefiks koji približno označava komponentu na koju se opcija odnosi. Obrazac je sledeći:

grupa\_imeOpcije

Većina ovih parametara zahteva duboko poznavanje wsrep. Objasnićemo samo najvažnije.

* base\_host: Ovo je IP adresa ili ime čvora.
* base\_port: Ovo je port korišćen za replikaciju.
* evs.inactive\_timeout: Ovo određuje koliko vremena čvor može biti nedostupan pre nego što se koristi težinski kvorum algoritam za rešavanje problema.
* evs.user\_send\_window: Ovo određuje broj paketa koji se mogu replicirati zajedno. Ako je mreža spora, ova vrednost treba da se poveća. Ova opcija je dinamična.
* gcache.dir: Ovo postavlja putanju za GCache fajlove. Omogućava pisanje GCache fajlova na drugom uređaju za optimizaciju performansi. Ova opcija nije dinamična.
* gcache.mem\_size: Ovo je maksimalna veličina GCache-a. Ova opcija nije dinamična.
* gcache.page\_size: Ovo je veličina stranice GCache-a. Ova opcija nije dinamična.
* gcs.fc\_master\_slave: Ovo se može postaviti na ON ako samo jedan čvor u klasteru može biti korišćen kao master, što znači da se svi klijenti povezuju samo na taj čvor. Ovo omogućava neke wsrep optimizacije. Ova opcija nije dinamična.
* gcs.max\_throttle: Ovo određuje koliko se može ubrzati priprema čvora, usporavajući normalnu replikaciju. Što je niža vrednost, više će biti usporena prenos stanja. Sa vrednošću 0, replikacija će se potpuno zaustaviti dok se prenos stanja ne završi. Zapamtite da u Galera klasteru, usporavanje replikacije za jedan čvor znači usporavanje replikacije za sve čvorove.
* pc.ignore\_quorum: Ovo onemogućava težinski kvorum algoritam. Postavljanjem ove vrednosti na 1 implicira da znamo kako da rešimo sukobe ako dođe do problema sa podeljenim mozgom.

# Monitoring i otklanjanje problema

Galera pruža set statusnih promenljivih koje se mogu koristiti za praćenje statusa svakog čvora. Kao i Galera server promenljive, svaka statusna promenljiva počinje sa prefiksom 'wsrep\_', što omogućava jednostavno vizualizovanje svih promenljivih pomoću sledećeg upita:

SHOW STATUS LIKE 'wsrep%';

Postoje najmanje dva načina za automatizaciju praćenja MariaDB Galera klastera:

* Korišćenjem dodatka nazvanog **Galera Cluster Nagios**. On je razvijen i održavan od strane FromDual-a.
* Korišćenjem **Galera automatskih obaveštenja**.

## Skripte za obaveštenja

Promenljiva servera wsrep\_notify\_cmd može se postaviti na shell komandu ili skriptu, koja će automatski biti pozvana kada čvor promeni svoj status. U pozivu će biti dodati određeni parametri kako bi se obaveštajna skripta opremila sa svim informacijama koje su joj potrebne. Skripta može koristiti ove informacije za obavljanje željenih akcija, kao što su beleženje događaja u datoteci, zapisivanje događaja u MariaDB tabelu ili slanje e-maila za obaveštavanje administratora baze podataka da je došlo do ozbiljnog problema.

Obaveštajna skripta će biti pozvana sa sledećom sintaksom:

<komanda> --status <novi\_status> --uuid <stanje\_UUID> --primary [yes|no] --members <lista\_članova> --index <indeks\_čvora>

Evo objašnjenja parametara:

* komanda: Ovo je komanda specificirana u promenljivoj wsrep\_notify\_cmd
* novi\_status: Trenutni status čvora
* stanje\_UUID: Univerzalno jedinstveni identifikator (UUID) koji je povezan sa poslednjom promenom stanja
* --primary: Ovaj argument ukazuje da li je čvor član primarnog klastera
* lista\_članova: Ovo je lista čvorova povezanih sa particijom klastera čvora
* indeks\_čvora: Indeks čvora

Da bismo Galeri rekli da koristi ovu skriptu, možemo koristiti sledeću izjavu:

SET @@global.wsrep\_notify\_cmd = '/putanja/do/notify.sh';

## Provera statusnih promenljivih

Periodično se može vršiti nekoliko provera kako bi se verifikovala celovitost klastera. Ove provere spadaju u četiri kategorije

* **Zdravlje klastera**: Želimo da saznamo da li je klaster particionisan i, ako jeste, da li su sve čvorove aktivne.
* **Zdravlje pojedinačnog čvora**: Za svaki čvor, želimo da proverimo da li je aktivan i, ako nije, potrebno je utvrditi razlog.
* **Zdravlje replikacije**: Čak i ako su svi čvorovi aktivni, želimo da proverimo da li je zakašnjenje replikacije prihvatljivo.
* **Performanse mreže**: Brzina komunikacije putem mreže

### Zdravlje klastera

Jedna od statusnih promenljivih koje želimo periodično proveravati jeste wsrep\_cluster\_size. Ako je ova vrednost manja od očekivane, to znači da je najmanje jedan čvor pao ili da je klaster podeljen na više particija. Dovoljno je proveriti ovu promenljivu na jednom čvoru. Međutim, ako uočimo da je veličina klastera preniska, želećemo da proverimo i ostale čvorove kako bismo utvrdili šta se desilo.

U tom slučaju, možemo proveriti i promenljive wsrep\_cluster\_state\_uuid i wsrep\_cluster\_conf\_id na svakom čvoru. Ove vrednosti su UUID-ovi koji u normalnim uslovima treba da budu identični za sve čvorove. Ako dva čvora imaju različite vrednosti za wsrep\_cluster\_state\_uuid, oni nisu povezani sa istim klasterom. Ako su te vrednosti identične, ali se razlikuje vrednost wsrep\_cluster\_conf\_id, klaster je podeljen na particije.

Ako je klaster podeljen, možemo se povezati sa jednim čvorom iz svake particije da bismo proverili koji od njih je primarni čvor. Da bismo to uradili, možemo upitati statusnu promenljivu wsrep\_cluster\_status. Provera ove promenljive za svaki čvor u klasteru još je jedan način da saznamo da li je klaster particionisan; ako barem jedan čvor ne pripada primarnom klasteru, postoji više od jedne particije.

### Zdravlje pojedinačnih čvorova

Ako je veličina klastera mala, možda ćemo želeti da proverimo svaki čvor da bismo utvrdili da li je u funkciji ili ne.

Najbolji način da proverimo da li je čvor ispravno pokrenut jeste da upitamo statusnu promenljivu wsrep\_ready. Ako je čvor u dobrom stanju, njegova vrednost treba da bude true. Ako nije, treba da pokušamo da utvrdimo šta nije u redu.

Statusna promenljiva wsrep\_connected pokazuje da li je wsrep biblioteka pokrenuta i povezana sa MariaDB. Ovo obično znači da wsrep nije mogao da se učita zbog greške u konfiguraciji. U tom slučaju, proverićemo ispravnost promenljivih poput wsrep\_cluster\_address.

Ako je vrednost wsrep\_connected true, možemo proveriti vrednost wsrep\_local\_state\_comment. Ako je "Joining" (Pridruživanje), "Waiting for SST" (Čekanje na SST) ili "Joined" (Pridružen), čvor je još uvek povezan sa klasterom. Sa velikim bazama podataka i sporim vezama, faza "Waiting for SST" može potrajati dugo vremena.

### Zdravlje replikacije

U sinhronoj replikaciji, klaster nije brži od najsporijeg čvora. To se dešava zato što svi čvorovi, nakon izvršenja transakcije, moraju da sačekaju da i svi ostali čvorovi završe istu transakciju. Zbog toga je važno periodično proveravati statusnu promenljivu wsrep\_flow\_control\_paused u Galera klasteru. Ova vrednost je u opsegu od 0 do 1 i predstavlja udeo vremena koje je čvor proveo čekajući da ostali čvorovi završe transakciju. Ako vrednost nije veoma blizu nuli, imamo problem sa kašnjenjem.

U tom slučaju, treba identifikovati sporije čvorove. To ćemo uraditi proverom dve statusne promenljive:

* wsrep\_flow\_control\_sent
* wsrep\_local\_recv\_queue\_avg

Sporiji čvorovi imaju više vrednosti.

Ako je čvor spor, trebalo bi pokušati povećati broj paralelnih niti koje se koriste za replikaciju. Ne zaboravite da proverite uobičajene probleme sa performansama MariaDB, poput lošeg korišćenja bafera memorije.

### Performanse mreže

Ako nijedan od čvorova nije primetno sporiji od drugih i još uvek nismo zadovoljni performansama, verovatno je usko grlo brzina mreže. Da bismo to proverili, možemo pogledati statusnu promenljivu wsrep\_local\_send\_queue\_avg. Spora mreža dovodi do visokog broja čekajućih poruka. Trivijalni, ali važan alat ping može potvrditi lošu mrežnu performansu. Takođe, iftop može pružiti podatke o saobraćaju na mreži, potvrđujući da li je propusni opseg zasićen.

Postoji mnogo mogućih razloga za sporu mrežu, a ova tema ne može biti potpuno pokrivena u knjizi o MariaDB-u. Neke opšte smernice su sledeće:

* Galera zahteva posvećenu podsredu mreže. Ostale komunikacije usporavaju replikaciju.
* Proverite konfiguraciju sistema, uključujući podešavanja softvera za zaštitu od požara.
* Ako je uzrok teško pronaći, treba razmotriti fizički sloj mreže. Ponekad možemo otkriti da je kabl previjen ili da je blizu magnetnog izvora. Elektromagnetna interferencija verovatno usporava mrežu ili je čini nepouzdanom.

## Balansiranje opterećenja

Optimizacija koja se može primeniti na bilo koji računarski klaster sastoji se u balansiranju zahteva među čvorovima, tako da svaki od njih ima približno isti nivo posla. Klijenti se povezuju sa balanserom opterećenja, koji deluje kao proxy pokušavajući ravnomerno preusmeriti komunikaciju. Postoji nekoliko balansera opterećenja, kako open source, tako i vlasničkih. Većina njih su generički balanseri, dizajnirani da rade sa bilo kojom vrstom komunikacije. Stoga se mogu koristiti za veb servere, fajl servere ili općenito za bilo koje vrste servera. Međutim, ne svi balanseri opterećenja dobro se ponašaju sa serverskim bazama podataka.

U ovom radu fokusiraćemo se na balanser opterećenja koji je specifično dizajniran za Galera klaster: Galera Load Balancer.

## Lista ograničenja Galera klastera

MariaDB Galera klaster ima nekoliko ograničenja zbog svoje dizajnirane strukture. Razvijači i DBA (Administratori baza podataka) trebali bi biti svesni da neke funkcije koje su dostupne na samostalnim MariaDB serverima nisu dostupne za Galera čvorove.

Prvo, kao što smo već spomenuli, Galera podržava samo operativni sistem Linux.

Galera je dizajnirana da se koristi sa InnoDB skladišnim mehanizmom. Eksperimentalna podrška za MyISAM postoji, ali je isključena prema zadatim postavkama. Da bi se omogućila, server promenljiva wsrep\_replicate\_myisam mora biti postavljena na 1. Tim Galera ohrabruje njeno korišćenje samo u eksperimentalne svrhe.

Nijedan drugi skladišni mehanizam ne može se koristiti sa Galerom.

Binarni zapis mora biti omogućen i njegov format mora biti ROW. Nije podržano bazirano na izjavi ili mešovito replikaciju.

Izjave koje stiču eksplicitne brave, kao što su SELECT … FOR UPDATE, SELECT … LOCK IN SHARE MODE, LOCK TABLES, ili FLUSH TABLES … FOR EXPORT nisu podržane. To dodatno smanjuje privlačnost MyISAM-a. Izolacioni nivo SERIALIZABLE, koji pretvara sve obične upite u upite zaključavanja, takođe nije podržan. Razlog tome je što operacije čitanja i bravice nisu širene preko klastera, pa se brave stiču samo na jednom čvoru, što može izazvati potencijalne konflikte. Postoji izuzetak: ako se svi klijenti povežu na isti čvor, brave su bezbedne.

Galera je dizajnirana da replikuje tabele koje imaju primarni ključ. Različiti problemi mogu se javiti sa tabelama koje nemaju eksplicitan primarni ključ, na primer, DELETE neće biti podržan, XA transakcije neće biti podržane, dupli InnoDB write bafer će biti onemogućen, i keširanje upita neće biti podržano.

Izjave koje eksplicitno modifikuju tabelu u mysql bazi podataka nisu podržane. Generalni upitni zapis i zapis grešaka, ako su omogućeni, ne mogu se pisati u sistemskim tabelama.

## Galera Load Balancer

Galera Load Balancer (GLB) je alat treće strane koji je razvila kompanija FromDual. Slično Galeri, GLB radi samo na Linux operativnom sistemu. Sastoji se od demona nazvanog glbd.

Ne postoji klijentska aplikacija za upravljanje GLB-om. Za slanje administrativnih komandi GLB-u, kao što je dodavanje ili uklanjanje čvorova, može se koristiti generički alat nc. Alat nc može komunicirati sa TCP daemonom i ispisati odgovor na ekran, što je u osnovi sve što nam je potrebno za upravljanje GLB-om. Evo opšte sintakse za slanje komande glbd sa komandne linije:

echo "<komanda>" | nc -q 1 <adresa\_hosta> <port>

Promenljiva adresa\_hosta je naziv hosta ili IP adresa na kojem se glbd pokreće, obično 127.0.0.1. Promenljiva port je port na koji glbd osluškuje. Ne postoji standardni port; moramo ga specificirati prilikom pokretanja demona.

Sintaksa za pokretanje glbd je sledeća:

glbd [OPCIJE] <port> <lista čvorova>

Promenljiva port može biti kompletan adresni prostor ako mašina ima više mrežnih interfejsa: address:port.

Promenljiva lista čvorova je lista čvorova Galera klastera razdvojena razmakom. Svaki čvor se može specificirati kao adresa:port:težina. Težina je važan koncept za glbd, ali se koristi samo ako je demon pokrenut sa opcijama --top ili --single, ili ako korišćena politika uzima u obzir težinu čvora. Sa opcijom --top, čvorovi sa većom težinom će uvek biti korišćeni ako barem jedan od njih radi. Ova opcija nema efekta ako svi pokrenuti čvorovi imaju istu težinu. Sa opcijom --single, koristi se samo jedan čvor sa najvećom težinom sve dok ne prestane sa radom. Težina čvora je korisna osobina ako neki serveri rade na mašinama sa ograničenim resursima i trebaju se koristiti samo za replikaciju, osim ako drugi čvorovi ne prestanu sa radom.

Podržane su standardne informativne opcije u glbd-u. Opcije --help i --version mogu se koristiti, redom, za ispis poruke pomoći ili broja verzije i zaustavljanje programa. Opcija --verbose se može koristiti za ispis dodatnih informacija na ekranu.

Neki od najvažnijih opcija su:

* Opcija --daemon pokreće glbd kao daemon.
* Opcija --control <port> specificira koji port će se koristiti za prihvatanje administrativnih komandi putem nc.
* Opcija --discovery omogućava autoodkriju novih čvorova kada su dodati u klaster. Lista čvorova se dobija iz poznatih čvorova.
* Opcije --top i --single nalažu glbd-u da uzima u obzir težinu čvora, kako je prethodno opisano.
* Opcija --max\_con <broj> postavlja maksimalan broj prihvaćenih veza kako bi se izbeglo preopterećenje klastera. Čak i ako ova opcija nije navedena, ograničenje je nametnuto od strane operativnog sistema.
* Opcija --threads <broj> specificira broj niti za upotrebu. Prema zadanim postavkama, koristi se samo jedna nit.
* Obično, glbd spaja male pakete podataka u veće pakete radi optimizacije upotrebe mreže. Opcija --nodelay onemogućava ovaj mehanizam.

Osima ako se koristi opcija --single, obično želimo da odredimo politiku koju će glbd koristiti za odabir odredišta za svaku SQL naredbu. Politika se može izabrati navođenjem odgovarajuće opcije. Sledeće politike su podržane:

* Najmanje povezano je podrazumevana politika koja se koristi kada nije navedena druga politika. Redirektuje svaku vezu ka čvoru koji je primio malo veza do sada. Težina čvora se takođe uzima u obzir, tako da će teži čvorovi primati više veza od lakših čvorova.
* U opciji --round, glbd koristi cirkularnu listu čvorova. Kada se zahtev za vezu primi, redirektuje se na trenutni čvor, a kursor napreduje ili se vraća na prvi čvor u listi.
* U opciji --random, svaka veza se redirektuje na slučajan čvor.
* U opciji --source, svaki klijent je dodeljen različitom serveru. Svi zahtevi za vezu od istog klijenta uvek će biti preusmereni na isti čvor, osim ako čvor ne prestane sa radom.

Evo nekoliko primera poziva glbd-a:

glbd --daemon --control 8765 --threads 4 3306 host1:4567:1 host2:4567:1 host3:4567:1

U ovom primeru, daemon će osluškivati port 3306 (standardni MariaDB i MySQL port) za klijentske veze, i port 8765 za administrativne komande. Imamo tri hosta sa istom težinom i koristi se standardna politika. Varijabla glbd koristi četiri istovremene niti.

glbd --daemon --single 3306 host1:4567:3 host2:4567:2 host3:4567:1

U ovom primeru, glbd radi na standardnom MariaDB portu ali ne osluškuje nijedan port za administrativne komande, pa neće biti moguće modifikovati listu čvorova tokom rada. Postoje tri čvora sa različitim težinama, ali koristi se samo host1. Ako host1 prestane sa radom, koristiće se host2, a ako taj čvor takođe prestane sa radom, glbd će koristiti host3.

U sledećem primeru, koristićemo nc za dodavanje novog hosta:

echo "host4:4567:1" | nc -q 1 127.0.0.1 8765

Demon bi trebalo da se pokreće na lokalnom računaru i osluškuje port 8765 za administrativne komande. Dodati čvor je node4. Njegova težina je 1.

U sledećem primeru pokazujemo kako izbrisati čvor iz liste:

echo "host2:4567:-1" | nc -q 1 127.0.0.1 8765

U prethodnom primeru postavljamo host2 sa negativnom težinom (-1). Negativne težine se koriste za brisanje servera iz liste, tako da host2 više neće biti korišćen.

Možemo takođe koristiti nc da dobijemo neke statistike korišćenja od demona:

echo "getinfo" | nc -q 1 127.0.0.1 8765

# Zaključak

Galera klaster za MariaDB donosi značajne prednosti koje nadmašuju tradicionalne standalone servere. U ovom seminarskom radu razmotrili smo kako postaviti i upravljati klasterom MariaDB servera.

Razgovarali smo o tome kako instalirati, konfigurisati i pokrenuti pojedinačne čvorove povezivanjem istih. Naučili smo kako koristiti Galera arbitra kako bismo se nosili sa situacijama gde jedna grupa čvorova izgubi vezu sa ostatkom klastera.

Diskutovali smo o tome kako pratiti klaster i identifikovati uzroke problema sa performansama.

Iako su čvorovi MariaDB Galera klastera veoma slični običnim MariaDB serverima, naučili smo o najvažnijim ograničenjima klastera.

Na kraju, upoznali smo se sa Galera Load Balancer-om, koji može biti korišćen za distribuciju opterećenja među čvorovima.

U zaključku, Galera klaster za MariaDB predstavlja izvanredan izbor za organizacije koje teže visokoj dostupnosti i pouzdanosti u upravljanju podacima. Njegove ključne prednosti leže u sinhronizovanoj replikaciji na nivou transakcija, što omogućava da se podaci konzistentno distribuiraju i ažuriraju među čvorovima u realnom vremenu. Ovaj pristup obezbeđuje minimalno vreme za oporavak od kvarova i gubitka podataka, čime se smanjuje rizik od prekida poslovanja. Takođe, Galera klasteri pružaju mogućnost horizontalnog skaliranja i efikasno upravljaju opterećenjem pomoću alata kao što je Galera Load Balancer. Kombinacija ovih karakteristika čini Galera klaster idealnim za okruženja koja zahtevaju visoku dostupnost, otpornost na kvarove i brzo reagovanje na promenljive zahteve korisnika i aplikacija.

# Literatura:

[1] [Galera Cluster for MySQL | The world's most advanced open-source database cluster.](https://galeracluster.com/)

[2] [MariaDB Galera Cluster - MariaDB Knowledge Base](https://mariadb.com/kb/en/galera-cluster/)

[3] [The Codership Documentation — Galera Cluster Documentation](https://galeracluster.com/library/documentation/)

[4] [Using Docker — Galera Cluster Documentation](https://galeracluster.com/library/documentation/docker.html)

[5] **MariaDB High Performance,**  Federico Razzoli