UNIVERZITET U NIŠU

ELEKTRONSKI FAKULTET

**Oporavak MongoDB baze podataka**

Seminarski rad iz predmeta

Sistemi za upravljanje bazama podataka

Student: Mentor:

Marta Đorđević, 1490 prof. dr Aleksandar Stanimirović

Niš, 2023. godina

Sadržaj

[Uvod 3](#_Toc136203619)

[Oporavak baze podataka 4](#_Toc136203620)

[Oporavak MongoDB baze podataka 6](#_Toc136203621)

[Pravljenje rezervnih kopija i vraćanje 6](#_Toc136203622)

[Korišćenjem MongoDB Atlasa 7](#_Toc136203623)

[Korišćenjem Filesystem Snapshot-ova 7](#_Toc136203624)

[Korišćenjem MongoDB alata 9](#_Toc136203625)

[Mehanizam replikacije 11](#_Toc136203626)

[Zaključak 21](#_Toc136203627)

[Literatura 22](#_Toc136203628)

# Uvod

MongoDB je jedan od najpopularnijih NoSQL (ne-relacionih) sistema za upravljanje bazom podataka koji je stekao ogromnu popularnost u poslednjoj deceniji. Njegova fleksibilnost, skalabilnost i sposobnost rukovanja velikim količinama podataka čine ga izuzetno korisnim za različite vrste aplikacija i poslovnih scenarija.

Ovaj seminarski rad ima za cilj da istraži različite aspekte oporavka MongoDB baze podataka. Obradićemo različite vrste problema s kojima se može suočiti MongoDB baza podataka, kao što su hardverski kvarovi, greške u softveru, gubitak podataka i slično. Takođe ćemo razmotriti metode i tehnike koje se mogu primeniti kako bi se obnovila funkcionalnost i integritet MongoDB baze podataka. Sigurnosno kopiranje je ključni deo procesa oporavka jer omogućava obnovu podataka u slučaju gubitka ili oštećenja baze podataka. Razmotrićemo kako pravilno planirati i izvršavati sigurnosna kopiranja.

# Oporavak baze podataka

Oporavak baze podataka je proces vraćanja baze podataka u stabilno stanje nakon kvara koji je doveo do gubitka i oštećenja podataka. Cilj ovog procesa je obnova podataka tako da se očuva njihova dostupnost, pouzdanost i celovitost.

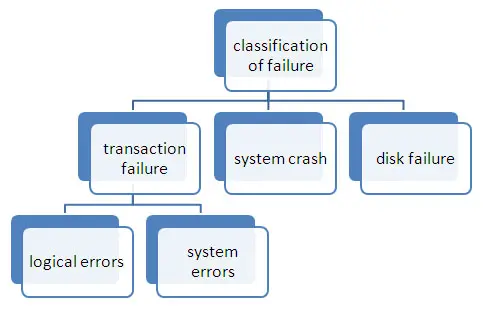
Kvarovi koji mogu zahtevati oporavak baze podataka uključuju hardverske kvarove, softverske greške, korisničke greške, napade hakera ili prirodne katastrofe. U ovim situacijama podaci mogu postati nedostupni ili oštećeni, što može dovesti do prekida poslovnih operacija ili gubitka vrednih informacija.

Tehnike oporavka baze podataka se koriste u sistemima za upravljanje bazama podataka za vraćanje baze podataka u konzistentno stanje nakon što je došlo do kvara ili greške. Oporovak baze podataka uključuje sledeće korake:

* Identifikacija problema: Prvi korak je utvrditi prirodu problema i odrediti uzrok gubitka ili oštećenja podataka. Ovo može uključivati pregled logova i analizu događaja.
* Sigurna kopija podataka: Za obnovu se koriste sigurne kopije podataka ako su dostupne. To su prethodno spremljeni snimci podataka koji se koriste za obnavljanje podataka u slučaju kvara.
* Obnova podataka: Uz pomoć gore navedenih sigurnih kopija, podaci se obnavljaju i vraćaju u bazu podataka. Ovaj korak takođe uključuje i vraćanje podataka na njihovo poslednje stanje pre kvara.
* Provera konzistentnosti: Nakon što se podaci obnove vrši se provera konzistentnosti kako bi se osiguralo da su obnovljeni podaci ispravni.
* Ponovno pokretanje: Kada su podaci uspešno obnovljeni, baza podataka se može ponovo pokrenuti i spremna je za dalju upotrebu.

U osnovi postoje sledeće vrste kvarova i grešaka koje se mogu desiti i dovesti do neuspešne transakcije:

* Neuspeh transakcije – transakcija treba da se prekine kada ne uspe da se izvrši ili kada dođe do dela da više be može dalje. Razlozi z a neuspeh transakcije su:
  + Logičke greške – kada se transakcija ne može izvršiti kao rezultat greške koda ili neke interne greške, kao što je deljenje nulom.
  + Sistemske greške – kada sistem prekine transakciju jer nije u stanju da je izvrši ili kada ima problem sa resursima.
* Pad sistema – spadaju hardverske, softverske ili mrežne greške.
* Otkazivanje diska – u prošlosti ovo je bila tipična greška. Kvarovi diska uključuju formiranje opasnih sektora, nedostupnost i pad diska ili drugi kvar koji će uništiti ceo ili deo skladišta diska.
* Ljudske greške – kao što su slučajno brisanje, ažuriranje ili netačan unos podataka što može dovesti do nedoslednosti podataka.
* Kršenje bezbednosti – bezbedonosna kršenja kao što su hakovanje ili neovlašćeni pristup što može ugroziti integritet podataka.



Slika 1 – Vrste kvarova

Tehnike koje se najčešće koriste za oporavak baze podataka:

* Tehnika rollback/undo – ova tehnika zasnovana je na principu povlačenja ili poništavanja efekta transakcije koja nije uspešno završena zbog kvara sistema ili neke greške. Ovo se postiže poništavanjem promena koje su izvršene u transakcijij koristeći logove koji se skladište u evidenciji transakcija koja sadrži zapise svih transakcija koje su izvršene u bazi podataka. Sistem ovi logove koristi kako bi poništio promene koje je izvršila neuspela transakcija i vratio bazu podataka u njeno prethodno stanje.
* Tehnika commit/redo – vrši se ponovno primene transakcije koja je uspešno izvršena u bazi podataka. Ovo se postiže korišćenjem zapisa evidencija kako bi se ponovo izvršile promene koje je obavila transakcija u toku greške. Sistem koristi zapise kako bi ponovo primenio promene koje je izvršila transakcija i kako bi vratio bazu podataka u njeno najnovije konzistentno stanje.
* Oporavak kontrolne tačke – tehnika koja se koristi za smanjenje vremena oporavka periodičnim čuvanjem stanja baze podataka. Ako se desi kvar, sistem može koristiti datoteku kontolne tačke da vrati bazu u konzistentno stanje koje je bilo pre nego što je došlo do greške, umesto da prolazi kroz ceo dnevnik da bi oporavio bazu podataka.

# Oporavak MongoDB baze podataka

MongoDB je document-oriented NoSQL baza podataka koja se koristi za skladištenje i manipulaciju podacima velikog obima. Za razliku od tradicionlanih relacionih baza podataka kod kojih se koriste tabele, MongoDB koristi kolekcije i dokumente i podaci se čuvaju u JSON (JavaScript Object Notation) formatu. Dokumenti se sastoje od parova ključ/vrednost. Kolekcije sadrže grupe dokumenata i slične su tabelama relacionih baza podataka. Svaki dokument može biti drugačiji, veličina i sadržaj polja mogu se razlikovati jedan od drugog.

Oporavak MongoDB baze podataka odnosi se na proces vraćanja baze podataka u ispravno stanje nakon nekog neočekivanog događaja koji može uzrokovati gubitak podataka ili oštećenje baze podataka. Kao što je prethodno rečeno, ovi događaji mogu da uključuju hardverske kvarove, softverske greške, padove sistema ili ljudske greške. Oporavak MongoDB baze podataka ima za cilj povratiti podatke u konzistentno stanje i to se postiže primenom različitih tehnika, a neke od njih će biti opisane u nastavku.

## Pravljenje rezervnih kopija i vraćanje

Pravljenje rezervnih kopija i vraćanje (Backup and Restore) u MongoDB-u predstavlja kreiranje kopija podataka i njihovo vraćanje kada je to potrebno. Na ovaj način se obezbeđuje sprečava gubljenje podataka, oporavak od grešaka i održava se integritet podataka.

Rezervne kopije su rezervne kopije koje kao tip podatka koriste BSON format. To je jedan od najvažnijih delova za svakog administratora baze podataka. Ona omogućava ponovno korišćenje baze podataka u slučaju da se desio bilo kakav gubirak. Postoje dve vrste rezervnih kopija: logičke rezervne kopije i fizičke rezervne kopije.

Logičke rezervne kopije kopiraju same podatke. Ne kopiraju nijednu fizičku datoteku koja se odnosi na podatke kao što su izvršne datoteke, datoteke evidencije ili kontrolne datoteke. Uglavnom se koriste za arhiviranje baza podataka i premeštanje baza podataka u različita okruženja i operativne sisteme. Podaci su šifrovani i zapisani kao .json, .bson ili .csv datoteka u zavisnosti od programa koji se koristi za pravljenje rezervnih kopija.

Fizičke rezervne kopije su kopije fizičkih datoteka koje pripadaju bazi podataka, kao što su datoteke sa podacima, kontrolne datoteke i datoteke evidencije. Ove datoteke se čuvaju na nekom tipu medijuma za skladištenje i mogu se koristiti za obnavljanje sistema baze podataka ukoliko bude bilo potrebe za tim.

Vraćanje MongoDB baze podataka iz rezervne kopije donosi osiguranje integriteta rezervne kopije, odabir odgovarajuće rezervne kopije i praćenje neophodnih koraka za vraćanje podataka u željeno stanje.

Načini na koje je moguće kreirati rezervne kopije i vratiti MongoDB bazu podataka:

### Korišćenjem MongoDB Atlasa



Slika 2 – Backup/Restore kod MongoDB Atlasa

### Korišćenjem Filesystem Snapshot-ova

Na ovaj način pravljenje rezervnih kopija se vrši pomoću alata na nivou sistema, kao što su LVM. Ovi filesystem snapshot-ovi ili ,,block-level” backupmetode, koriste alate na nivou sistema za kreiranje kopija uređaja koji drži fajlove podataka MongoDB-a.

Snapshot-ovi kreiraju pokazivače između trenutnih podataka i posebnih podataka snapshot-a. Ovi pokazivači su teoretski ekvivalentni sa „hard-links“. Pošto se podaci razlikuju, proces kreiranja snapshot-ova koristi strategiju „copy-on-write“ i kao rezultat toga snapshot čuva samo izmenjene podatke. Nakon kreiranja snapshot-a, snapshot image se mount-uje na file system i kopiraju se podaci sa snapshot-a. Dobijena rezervna kopija sadrži potpunu kopiju svih podataka.

MongoDB 3.2 je dodao podršku za rezervne kopije MongoDB instanci na nivou uređaja koristeći **WiredTriger** mehanizma za skladištenje kada se datoteke podataka MongoDB instance i datoteke dnevnika nalaze na različitim uređajima. Da bi se napravila koherentna rezervna kopija, baza podataka mora biti zaključana i svi upisi u bazu podataka moraju biti suspendovani tokom pravljenja rezervne kopije.

Snapshot-ovi kreiraju sliku celog diska. Ako nije potrebno praviti rezervnu kopiju celog sistema, bilo bi dobro izolovati datoteke MongoDB podataka, dnevnika i konfiguracije na jednom logičkom disku koji ne sadrži nikakve druge podatke.

Ukoliko mongod instanca ima omogućeno vođenje evidencije dnevnika, onda se može koristiti bilo koja vrsta file system-a ili snapshot alat na nivou uređaja/bloka za kreiranje rezervne kopije. Ako se backup-uje na Linux sistemu, neophodno je kofigurisati sistem uz pomoć LVM-a (Logical volume manager), da se disku omoguće paketi za mogućnost kreiranja snapshot-ova. LVM je softverski alat koji se koristi u Linux operativnom sistemu za upravljanje skladištenjem podataka. Omogućava korisnicima da upravljaju grupama fizičkih diskova kao jedinstvenim logičkim volumenima, pružajući veću fleksibilnost i skalabilnost u organizaciji skladišnih resursa. U nastavku će biti dat primer kreiranja rezervnih kopija i oporavka podataka korišćenjem LVM-a.

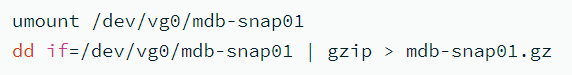
Za pravljenje snapshot-a korišćenjem LVM-a koristi se sledeća komanda:



Slika 3 – Kreiranje snapshot-a

Ova komanda krenira LVM snapshot pod nazivom *mdb-snap01*, MongoDB uređaja i pripada *vg0* grupi uređaja. Snapshot ima ograničenje od 100MB i predstavlja količinu razlike ozmeđu trenutnog stanja i kreiranja snapshot-a.

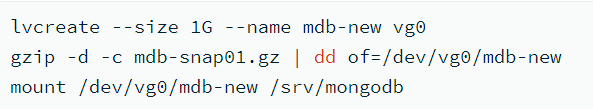
Nakon što je snapshot kreiran, potrebno je mount-ovati ga i kopirati podatke u zasebno skladište. Moguće je napraviti kopiju snapshot image na novou bloka sledećom komandom:



Slika 4 – Arhiviranje snapshot-a

Prva naredba osigurava da uređaj nije montiran, jer nikada ne treba praviti rezervne kopije na nivou bloka file system-a ili snapshot system-a kada je montiran. Druga naredba vrši kopiranje na nivou bloka celog snapshot image-a koristeći *dd* komandu i kompresuje rezultat u zipovani fajl u trenutnom radnom direktorijumu.

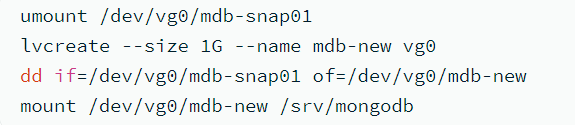
Restore snapshot-a kreiranog pomoću LVM-a postiže se sledećim komandama:



Slika 5 – Restore snapshot-a

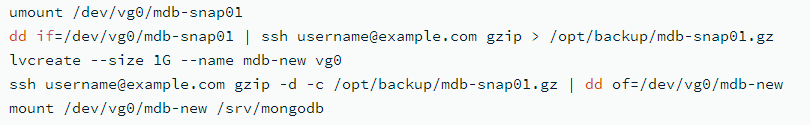
Prvom naredbom se kreira novi logički disk pod nazivom *mdb-new*, u */dev/vg0* grupi logičkih diskova. Druga naredba predstavlja proces dekompresovanja i smeštanja na *mdb-new* disk *image*, Poslednja naredba *mount*-uje disk *image* na */srv/mongodb* direktorijum.

Da bi se vratila rezervna kopija bez upisivanja u kompresovanu datoteku, mogu se koristiti sledeće naredbe:



Slika 6 – Restore snapshot-a bez upisivanja u zip fajl

Postoji i način implementacije rezervne kopije van sistema, odnosno kreiranje udaljene kopije koristeći kombinovane procese i SSH. Ove naredbe identične su prethodnim, osim što arhiviraju i kompresuju rezervnu kopiju na udaljenom sistemu koristeći SSH.



Slika 7 – Remote restore backup

### Korišćenjem MongoDB alata

**Mongodump** je alatka za pravljenje rezervnih kopija baze podataka i može izvoziti podatke iz samostalnih skupova replika i deljenih klastera. Pruža mogućnost kreiranja rezervne kopije celog servera, baze podataka, kolekcije ili dela kolekcije.

Nad svakom *mongodump* naredbom mogu se navesti *username* i *password* za autentifikaciju baze podataka

Da bi se povezalo sa lokalnom MongoDB instancom koja radi na portu 27017 i koristila podrazumevana podešavanja za izvoz sadržaja, potrebno je navesti samo *mongodump.*

Ovom naredbom sa argumentima može se specificirati hosti port MongoDB instance na koju *mongodump* treba da se poveže.



Slika 8 – Kreiranje rezervne kopije

Kreiraće se BSON fajl koji sadrže kopiju podataka dostupnih preko mongod slušanjem na portu *27017* hosta *mongodb.example.net.*

Da bi se naveo drugi izlazni direktorijum, može se koristiti *--out ili -o*



Slika 9 – Kreiranje rezervne kopije specificiranjem izlaznog direktorijuma

Da bi se ograničila količina podataka, postoje opcije *–db* i *–collection*:



Slika 10 – Kreiranje rezervne kopije sa ograničenom količinom podataka

Ova operacija kreira dump kolekcije pod nazivom *myCollection* iz baze podataka *test* u poddirektorijumu *dump/* trenutnog radnog direktorijuma.

*mongodump* sa opcijom *–oplog* skuplja oplog ulaze i omogućava pravljenje rezervne kopije podataka nad trenutnim podacima. Kada se kasnije restore*-*uje baza podataka iz kopije, baza podataka biće ista kao što je bila pre završetka pravljenja rezervne kopije.

**Mongorestore** je alatka za vraćanje baze podataka. Podrazumevano traži rezervnu kopiju u *dump*/ direktorijumu. On se direktno povezuje na pokrenuti mongod i može vratiti celu rezervnu kopiju baze podataka ili njen podskup.

Da bi se koristio mongorestore potrebno je povezati se na aktivni mongod komandom:



Slika 11 – Povezivanje mongorestore na aktivni mongod

*mongorestore* importuje rezervnu kopiju baze podataka u folder */opt/backup/mongodump-1* na mongod instanci koja je pokrenuta na localhost-u na podrazumevanom portu 27017.



Slika 12 – Specificiranje izlazbog direktorijuma

Ako želimo da definišemo drugi hosti port, to možemo učiniti na sledeči način:



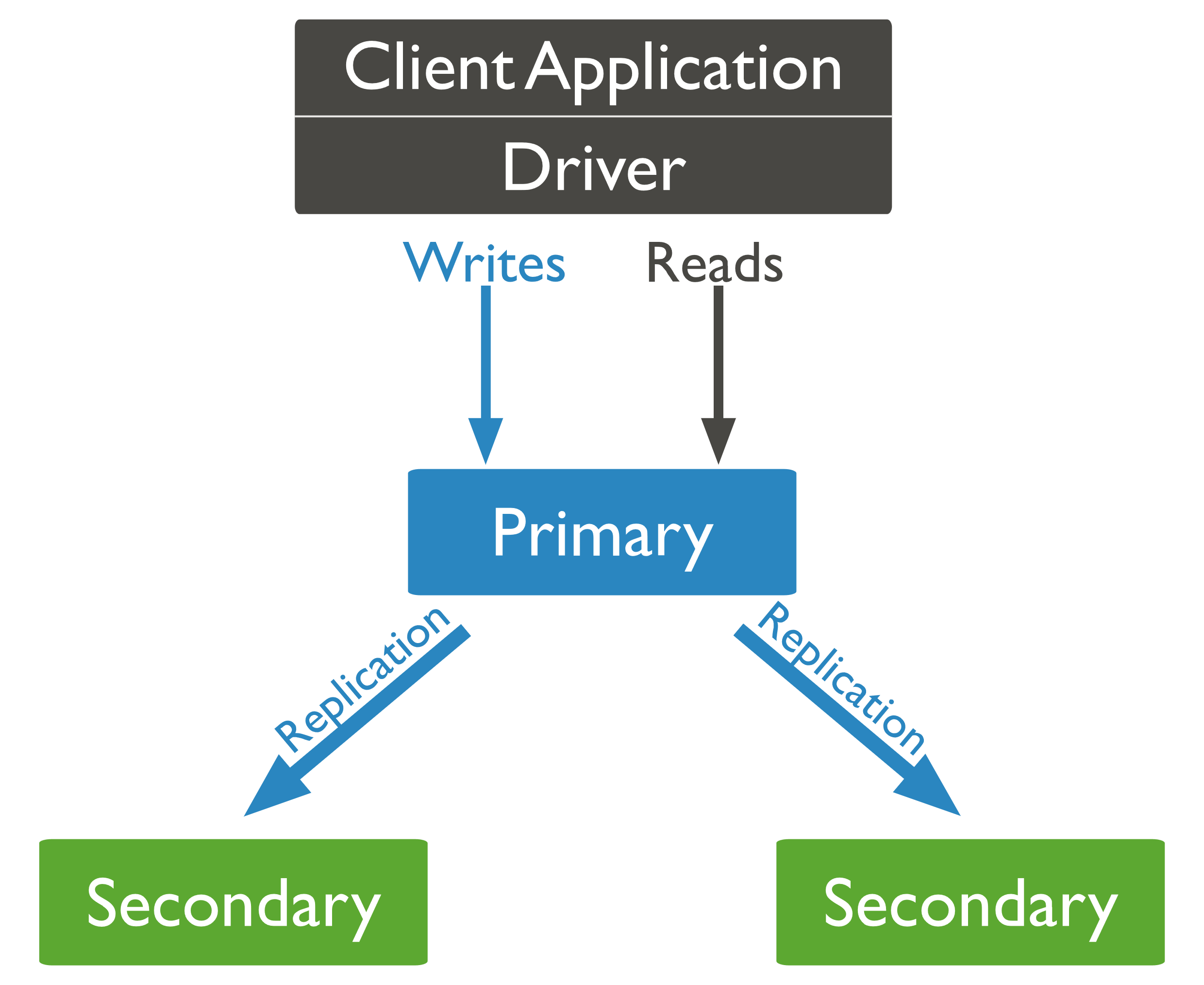
Slika 13 – Povezivanje mongorestore na udaljeni mongod

## Mehanizam replikacije

MongoDB mehanizam replikacije pruža sinhronizaciju podataka između većeg broja MongoDB instanci. Replikacija omogućava da se poveća dostupnost podataka na taj način što se kreiraju više kopija podataka na serverima. Skup replika je grupa mongod procesa koji održavaju isti skup podataka. Jedna instanca je primarna, a sve ostale su sekundarne. Svi klijenti upisuju podatke u primarnu instancu, a podaci se nakon toga asinhrono sinhronizuju sa sekundarnim instancama. Replikacija obezbeđuje redudantnost i povećava dostupnost podataka kod čitanja. Trajnost podataka se održava tako što se čuva više kopija ili replika tih podataka na fizički izolovanim serverima.

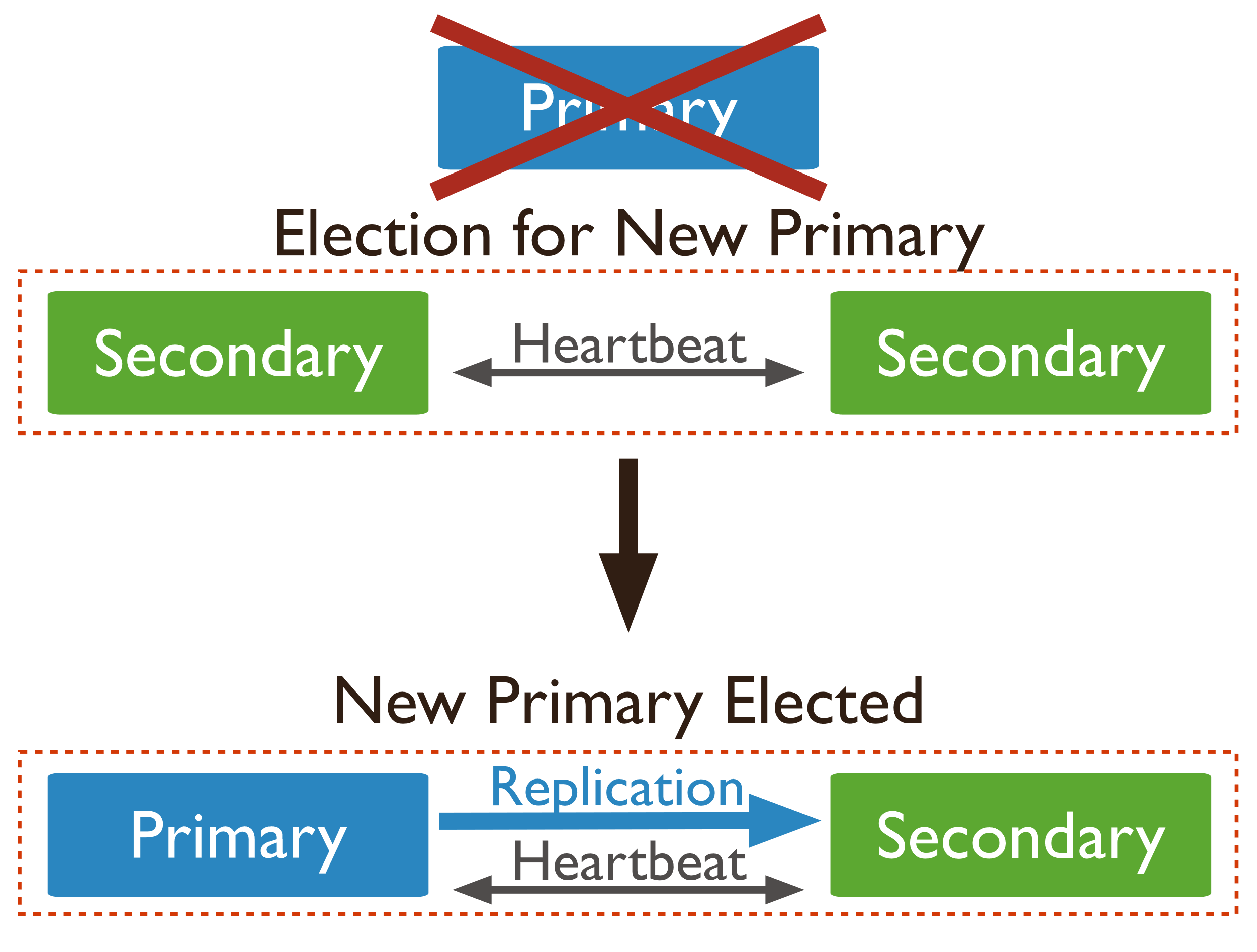
**Skup čvorova (replikacijski skup)** sadrži nekoliko čvorova koji imaju podatke i opciono jedan arbitarski čvor koji ne poseduje podatke. Od čvorova koji imaju podatke samo jedan može da bude primarni, dok se ostali čvorovi smatraju sekundarnim čvorovima. MongoDB obezbeđuje podršku za failover mehanizam. U slučaju da primarni čvor otkaže, mehanizmom glasanja ostale instance proglašavaju jednu od sekundarnih za novu primarnu.

**Primarni čvor** je jedini čvor u replikacijskom skupu koji podržava operacije pisanja (upite za dodavanje, ažuriranje i brisanje podataka) i on održava glavnu kopiju podataka, dok operaciju čitanja podržavaju svi članovi skupa replika. MongoDB podatke upisuje na primarni čvor, a zatim se operacije snimaju na oplogu primarnog čvora. Skup replika može imati najviše jedan primarni čvor, i ako on iz bilo kog razloga prestane sa radom određuje se novi primarni čvor.



Slika 14 – Mehanizam replikacije

**Sekundarni čvorovi** su replicirane kopije podataka sa primarnog čvora. Minimalni broj sekundarnih čvorova koje bi trebalo da ima skup replika jeste dva. Ako trenutni primarni čvor otkaže, metodom izbora bira se sekundarni čvor koji će preuzeti ulogu primarnog čvora i na taj način se sprečava gubljenje podataka usled kvarova i otkaza.

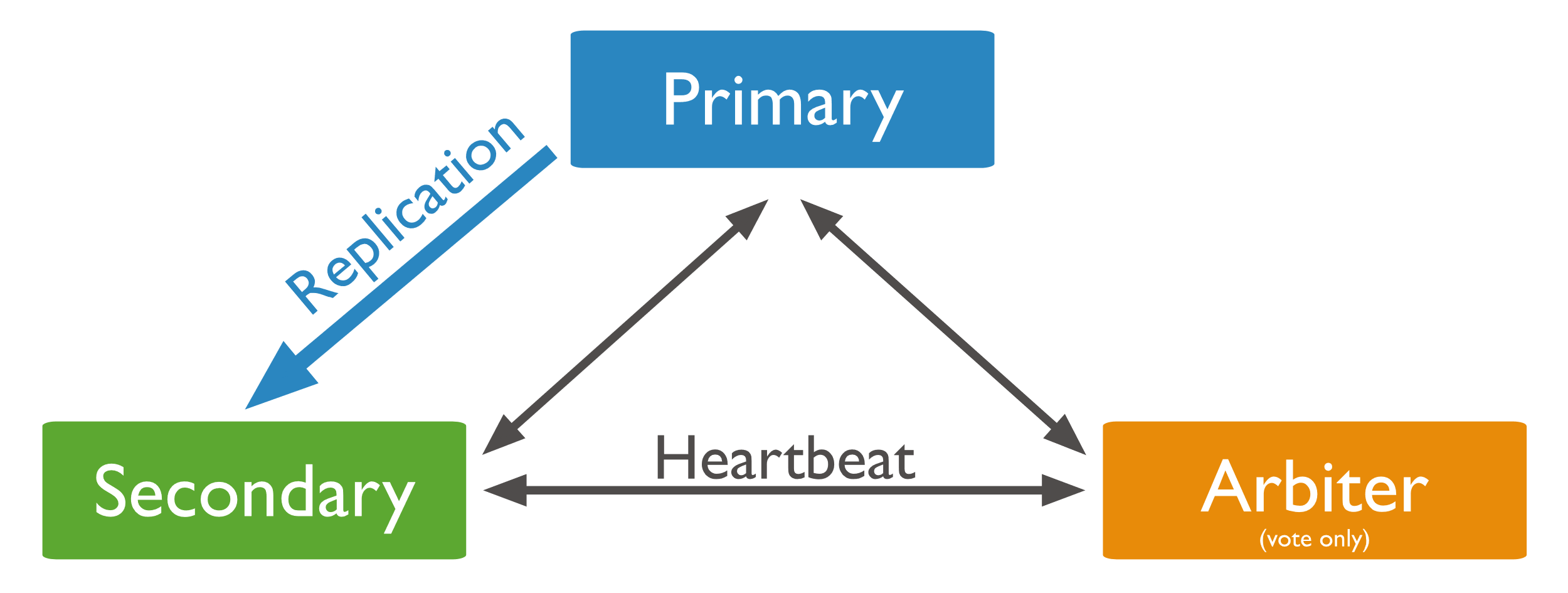


Slika 15 – Otkaz primarnog čvora

Proces otkucaja srca je proces kojim se identifikuje trenutni status čvora u skupu replika. Na svake dve sekunde čvorovi u skupu replika šalju pingove jedan drugome i u slučaju da neki čvor u roku od 10 sekundi ne pošalje ping nazad ta replika će se označiti kao nepristupačna. Ovo je od velikog značaja za proces automatskog pronalaska greške i nakon što se uoči greška neki od sekundarnih čvorova će postati primarni.

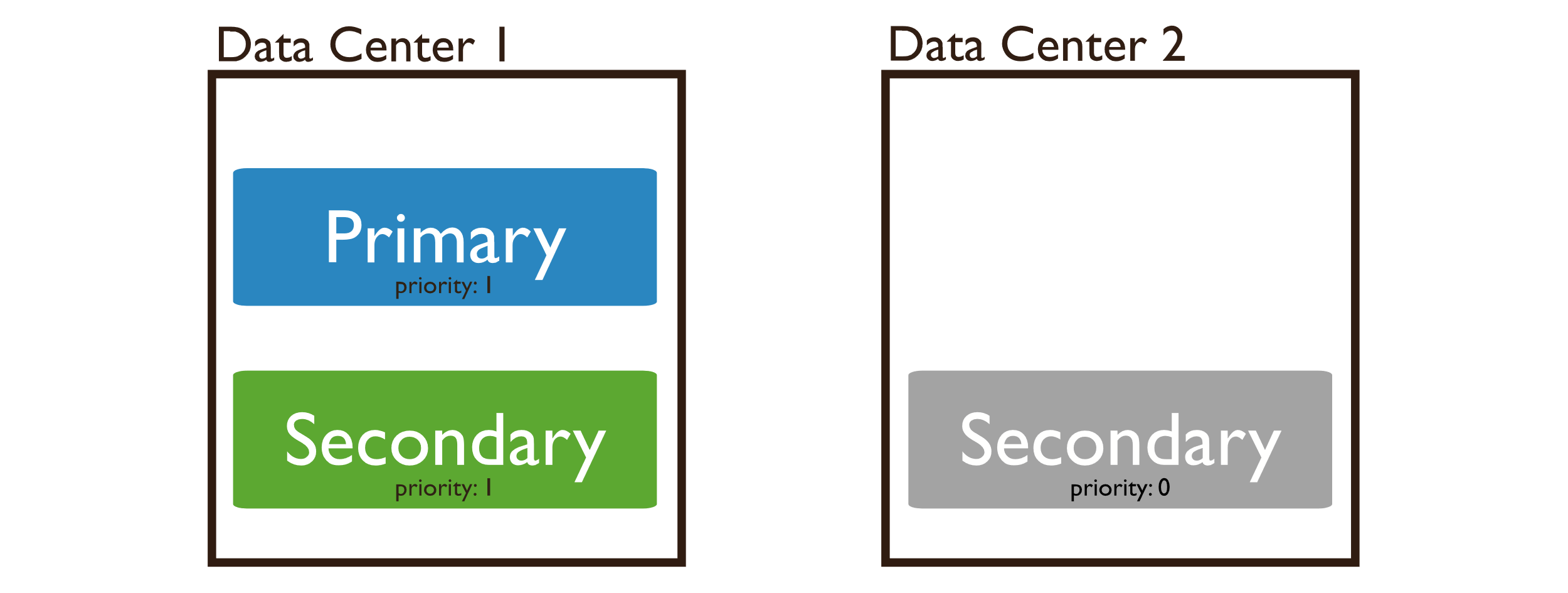
**Oplog (dnevnik operacija)** je posebna ograničena kolekcija koja vodi evidenciju o operacijama koje menjaju podatke uskladištene u bazama podataka. MongoDB primenjuje operacije baze podataka na primarnom čvoru, a zatim te operacije zapisuje na oplogu primarnog. Nakon toga ove operacije se kopiraju i preminjuju na sekundarnim čvorovima u asinhronom procesu. Svi članovi skupa replika sadrže kopiju oploga i to im omogućava da održavaju trenutno stanje baze.

**Arbitarski čvor** učestvuje na izborima za primarni čvor u slučaju da postojeći primarni čvor postane nedostupan, ali nema kopiju skupa podataka i ne može postati primarni, već samo obavlja funkciju glasača. Takođe doprinosi održavanju broja glasova potrebnih za glasanje kako bi se osigurala stabilnost i doslednost u izboru primarnog čvora. Kada ne bi postojao ovaj čvor, broj sekundarnih replika bi morao biti neparan kako bi se dobila većina. Pomoću njega se štede resursi jer ne zahteva dodatni prostor na disku potreban za replikaciju podataka jer ne sadrži kopije podataka, te postojanje arbitarskog čvora ne povećava opterećenje sistema. Uloga ovog čvora ne povećava otpornost na kvarove niti povećava dostupnost podataka jer se ne radi o kopiji, već samo pomaže u izboru novog primarnog čvora.



Slika 16 – Arbitarski čvor

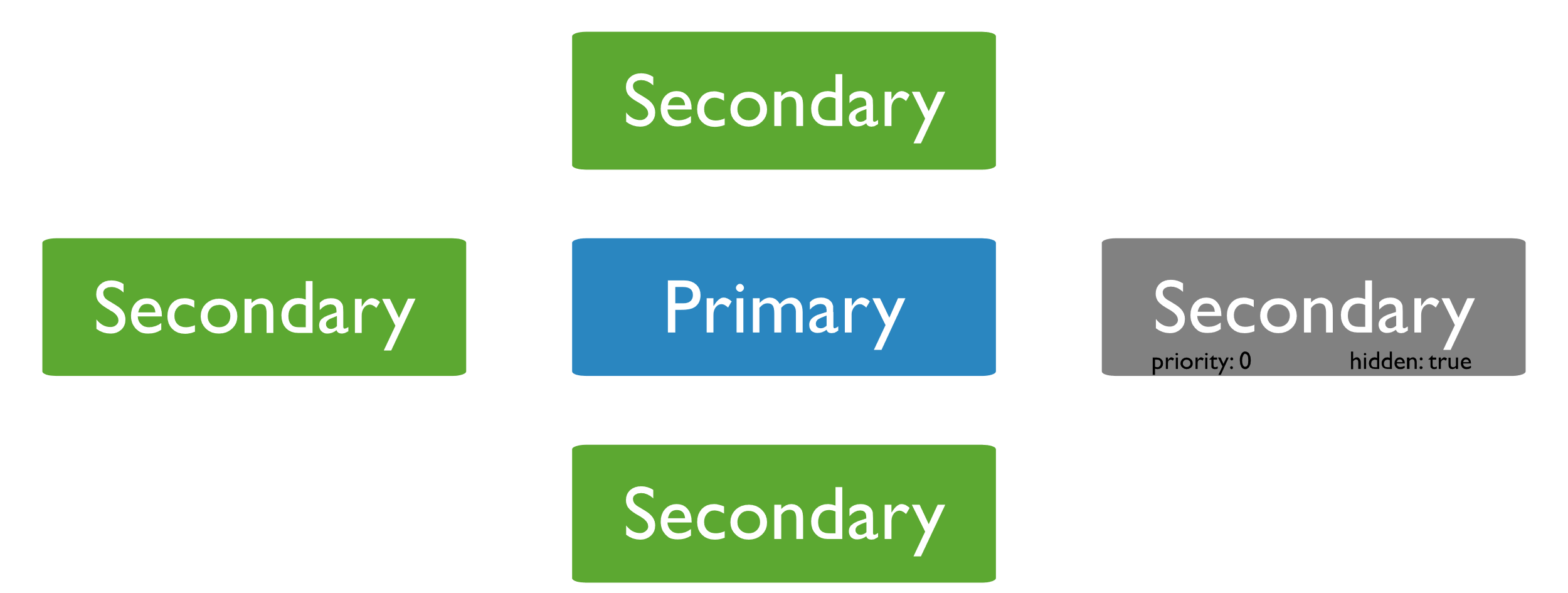
U MongoDB replikaciji postoje se **sekundarni čvorovi sa prioritetom 0**. To znači da taj čvor neće biti kandidat u izboru za primarni čvor, ne može postati primarni i ne može pokrenuti glasanje. Izuzev ovoga, oni funkcionišu kao normalni sekundarni čvorovi koji imaju kopije podataka, podržavaju operacije čitanja i glasaju na izborima. Čvorovi sa prioritetom 0 se uvode kada postoje članovi koji se nalaze u centru podataka koji je udaljen od glavne primene i zbog toga ima veće kašnjenje. Ovakvi čvorovi odgovaraju lokalnim zahtevima za čitanje, ali nisu dobri da obavljaju ulogu primarnog čvora.



Slika 17 – Sekundarni čvor sa prioritetom 0

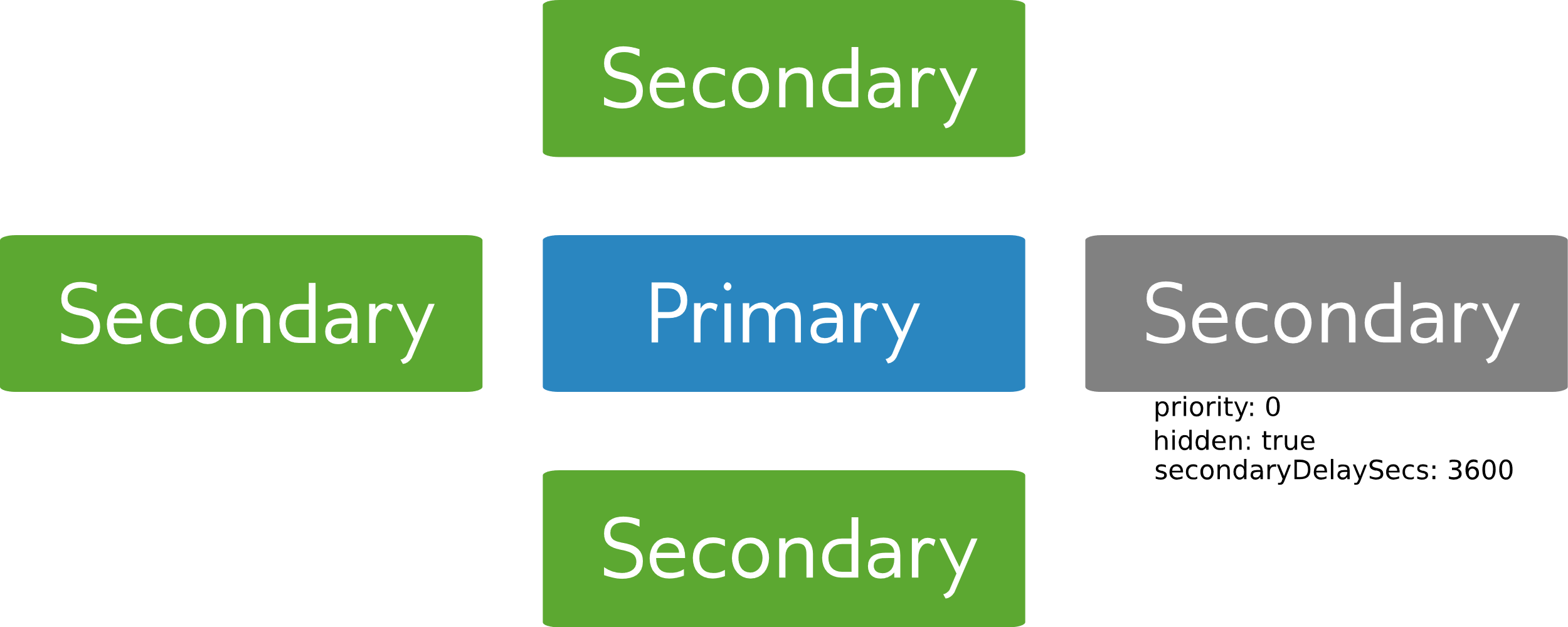
Na slici su prikazana dva centra podataka i samo sekundari iz levog centra mogu postati primarni čvor. Sekundarni čvor sa prioritetom 0 može služiti kao zamena ako u određenom vremenskom periodu nije moguće dodati novog člana u skup replika.

**Skrivene sekundarne replike** su poseban tip sekundarnih čvorova koji su konfigurisani tako da nisu vidljivi klijentskim aplikacijama za izvršenje operacija čitanja. Oni uvek moraju imati prioritet 0 na taj način ne mogu postati primanri čvorovi. Time što su nevidljivi za klijentske zahteve smanjuje se opterećenje replikacijskog skupa i poboljšavaju performanse. Iako su skrivene, one i dalje učestvuju u replikaciji i glasanju i često se koriste za obavljanje analitičkih operacija koje ne utiču na performanse primarnog čvora i klijentskih aplikacija.



Slika 18 – Skriveni sekundarni čvor

**Odložene sekundarne replike** su tip sekundarnih replika koje su konfigurisane tako da namerno kasne u primanju operacija pisanja od strane primarnog čvora i replikaciji. Ovo kašnjenje omogućava vraćanje skupa replika u prethodno stanje pre određenih operacija i grešaka. Kada se čvor konfiguriše kao odloženi, postavlja se vremensko kašnjenje između trenutka kada primarni čvor prima pisane operacije i trenutka kada se ne operacije kopiraju na odloženu sekundarnu repliku, pa odložena replika neće odmah imati promene sa primarnog čvora. Odložene sekundarne replike mogu pružiti žaštitu od zlonamernih napada i neovlašćenih promena podataka. U tom slučaju administrator može pomoću odložene replike povratiti integritet podataka i vratiti prethodne verzije.



Slika 19 – Odložena sekundarna replika

Izbor skupa replika se koristi da bi se odredilo koji čvor će postati primarni čvor. Izbori se dešavaju u sledećim situacijama:

* Gubitak veze sa primarnim čvorom koji se otkriva procesom otkucaja srca
* Inicijalizacija skupa replika
* Dodavanje novog čvora već postojećem skupu replika

Tokom izbora prvi čvor zahteva izbore, a svi ostali čvorovi glasaju kako bi se izabrao novi primarni čvor. Prosečno vreme za završetak ovog procesa je 12 sekundi. Ključni faktor koji može uticati na vreme koje je potrebno da se izvrše izbori je kašnjenje. Sve dok se izbori ne završe, operacija pisanja nije moguća, ali ako je upit za čitanje konfigurisan da se obrađuje na sekundarnom čvoru, operaciju čitanja je moguće izvršiti.

Kašnjenje u replikacijij je kašnjenje u kopiranju podataka na sekundarni čvor nakon ažuriranja primarnog. U nastavku će biti navedeno nekoliko faktora koji mogu povećati kašnjenje:

* Kašnjenje mreže – ako mreža ne može ispuniti potrebe za procesom replikacija, uzrokovaće kašnjenje u replikacijaja na ceo skup podataka, te je dobra praksa usmeriti saobraćaj na stabilnu mrežu sa dovoljnom propusnošću
* Propusnost diskova – ako se razlikuju diskovi koji se koriste na primarnom i sekundarnom čvoru, to može dovesti do kašnjenja u pisanju na sekundarni čvor
* Velika opterećenja – izvršavanje teških i dugotrajnih operacija pisanja na primarnom čvoru
* Operacije baze podataka – nekim upitima je potrebno više vremena da se izvrše

Razlozi zbog kojih je dobro korišćenje mehanizma replikacije:

* Visoka dostupnost podataka – ako primarni čvor prestane sa radom, sekundarni čvorovi mogu preuzeti funkciju primarnog čvora i osigurati dostupnost baze podataka
* Automatski oporavak od kvara – kada se primarni čvor oporavi od kvara, automatski će se sinhronizovati sa drugim čvorovima i vratiti funkciju primarnog čvora
* Bolja zaštita od gubitka podataka – replikacija osigurava da se podaci automatski kopiraju na druge čvorove u replikaciji i time se smanjuje rizik od gubitka podataka

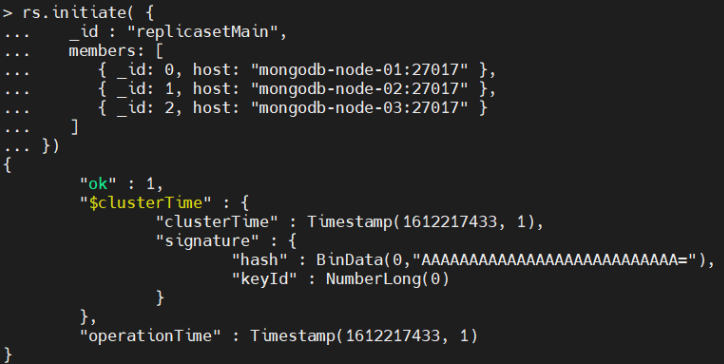
Da bi se konfigurisala replikacija potrebno je izvršiti sledeće naredbe:

Pokretanje 3 MongoDB procesa, prvi proces biće primarni, a drugi i treći će biti sekundarni, korišćenjem komande



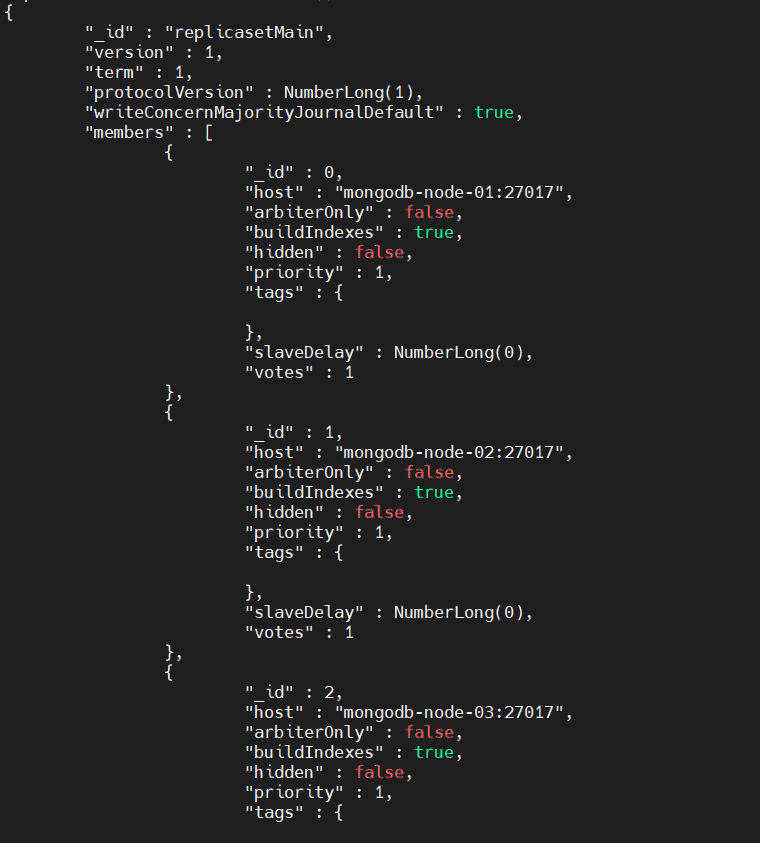
Povezivanje sa jednim od MongoDB procesa.

Moguće je inicijalizovati skup replika pomoću metode *rs.initiate().* Ovaj metod je potrebno izvršiti samo na jednoj MongoDB instanci u skupu replika. Unutar metode pokretanja, može se navesti ime skupa replika i člana.



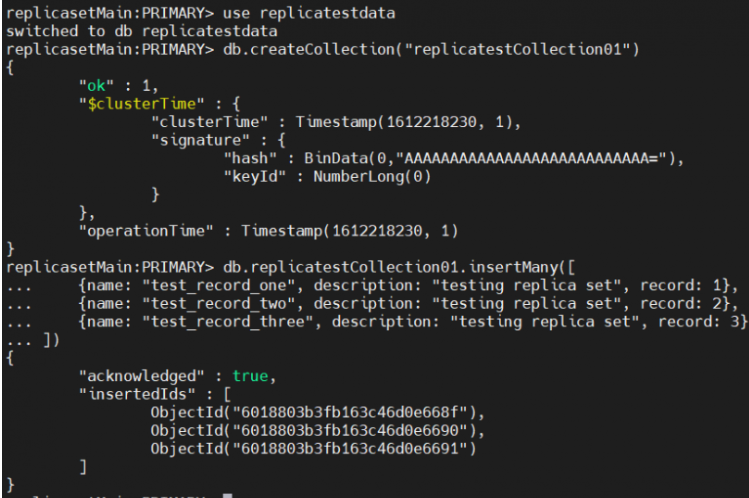
Slika 20 – Inicijalizacija skupa replika

Koristeći komandu *rs.conf(),* može se videti konfiguraciju skupa.



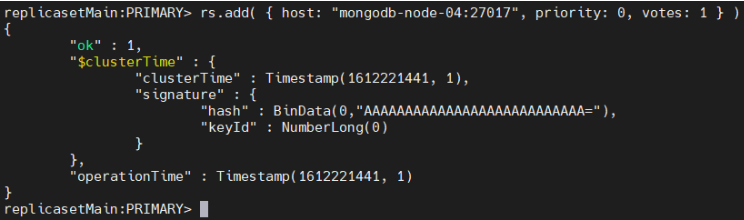
Slika 21 – Rezultat naredbe conf()

Sledeći korak je potvrda procesa replikacije. Prvo se prijaviti na primarni MongoDB čvor u skupu replika. Zatim morate da kreirate kolekciju sa nekim primerima podataka koristeći sledeće komande:



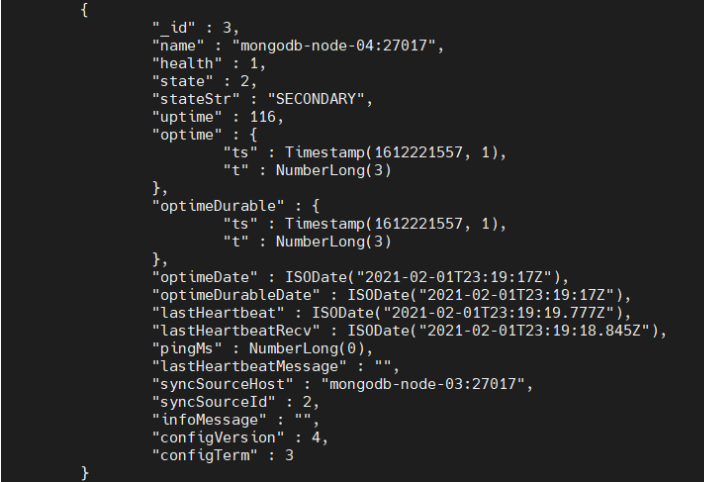
Zatim je potrebno na sekundarnim čvorovima proveriti da li su podaci uspešno replicirani.

Koristeći komandu *rs.add()*, dodaje se novi čvor postojećem skupu replika:



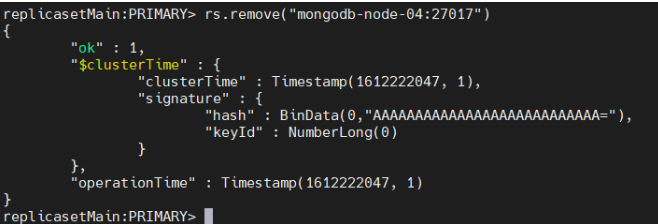
Slika 22 – Dodavanje novog čvora skupu replika

Moguće je proveriti da li je novi čvor dodat korišćenjem komande *rs.status()*, koja će prikazati detalje o novom čvoru.



Slika 23 – Provera statusa

Komanda *rs.remove()* se može koristiti za uklanjanje čvora iz skupa replika. Potrebno je isključiti instancu servera pre nego što se pokuša uklanjanjea čvor. Prilikom uklanjanja može se odrediti koji čvor treba ukloniti koristeći ime tog čvora.



Slika 24 – Uklanjane čvora iz skupa replika

# Zaključak

Možemo zaključiti da je oporavak MongoDB baze podataka ključni proces za održavanje integriteta podataka i obezbeđivanje kontinuirane dostupnosti aplikacija koje se oslanjaju na ovaj sistem za upravljanje bazom podataka. Različite vrste problema i grešaka mogu uticati na MongoDB bazu podataka, ali uz odgovarajuće metode i tehnike oporavka, moguće je povratiti funkcionalnost i integritet sistema.

Arhitektura replikacije pruža visok stepen dostupnosti i otpornosti na kvarove, omogućavajući održavanje više kopija podataka na različitim serverima. Oporavak nakon neočekivanog zaustavljanja servera zahteva odgovarajuće procedure za ponovno pokretanje i obnovu baze podataka. Gubitak podataka takođe zahteva posebne strategije i alate za povratak izgubljenih podataka. Sigurnosno kopiranje baze podataka igra ključnu ulogu u procesu oporavka, omogućavajući povratak podataka u slučaju gubitka ili oštećenja baze podataka. Pravilno planiranje, izvršavanje i održavanje sigurnosnih kopija su od vitalnog značaja za obezbeđivanje pouzdanog oporavka.

# Literatura

<https://www.mongodb.com/docs/manual/tutorial/backup-with-filesystem-snapshots/>

<https://www.mongodb.com/docs/manual/tutorial/backup-and-restore-tools/>

<https://hevodata.com/learn/mongodb-replication/>

<https://www.mongodb.com/docs/manual/replication/>