Univerzitet u Nišu

Elektronski fakultet

Katedra za računarstvo

*Replikacija i mirroring u MSSQL-u*

****

**Mentor:** **Student:**

Prof. dr Aleksandar Stanimirović Jelena Tošić, 1116

Niš, 2021. godine

**Sadržaj:**

[1. Uvod 1](#_Toc74915783)

[2. Replikacija u MSSQL-u 2](#_Toc74915784)

[2.1. Osnovni pojmovi i definicje u vezi sa replikacijom 3](#_Toc74915785)

[2.2. Tipovi replikacije u SQL Serveru 6](#_Toc74915786)

[2.3. Primer koraka prilikom kreiranja komponenata transakcione replikacije u SQL Serveru 10](#_Toc74915787)

[2.4. Primeri kreiranja i pregleda članaka kod transakcione replikacije SQL Servera 12](#_Toc74915788)

[2.5. Primeri preuzimanja informacija o ostalim komponentama replikacije 15](#_Toc74915789)

[3. *Mirroring* u SQL Serveru 18](#_Toc74915790)

[3.1. Operativni modovi *mirroring* baze 20](#_Toc74915791)

[3.1.1. Otkaz servera u modu visoke sigurnosti sa automatskim preusmeravanjem 22](#_Toc74915792)

[3.2. Podešavanje database mirroring okruženja u SQL Serveru 24](#_Toc74915793)

[3.2.1. Provera konfiguracije *endpoint*-a 26](#_Toc74915794)

[3.3. Razlike između replikacije i mirroring-a 27](#_Toc74915795)

[4. Zaključak 29](#_Toc74915796)

[5. Literatura 30](#_Toc74915797)

# Uvod

*Mirroring* i replikacija predstavljaju ključne koncepte koji su usko povezani sa procesom kopiranja podataka u okviru DBMS-a, pri čemu služe da povećaju dostupnost i performanse podataka odnosno baze podataka. Dostupnost baze je presudna za različita preduzeća i velike kompanije, čineći sigurnosnu kopiju (engl. backup) i replikaciju baze podataka neophodnom tehnikom za očuvanje podataka u raznim sverama poslovanja.

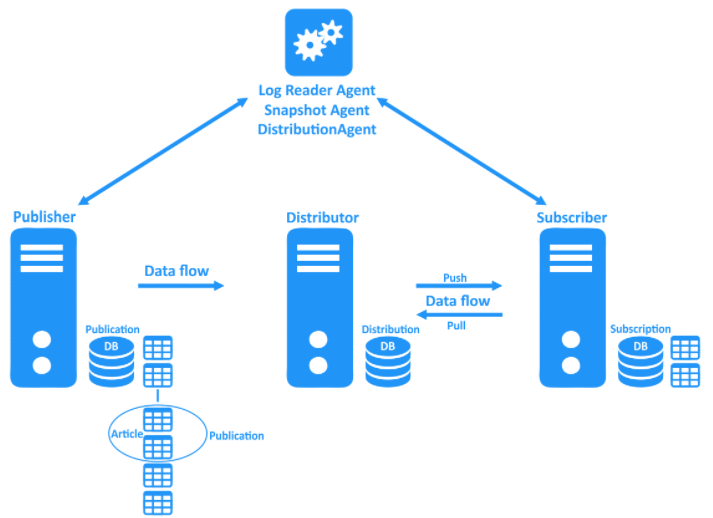
Sa MSSQL Server replikacijom moguće je stvoriti identičnu kopiju primarne baze podataka ili sinhronizovati promene između više baza podataka i održavati konzistentnost i integritet podataka. Sa druge strane, *mirroring* predstavlja tehniku koja se može koristiti zajedno sa replikacijom da bi se poboljšala dostupnost baze podataka. *Mirroring* baze podataka uključuje dve kopije jedne baze podataka koje se obično nalaze na različitim računarima, pri čemu je u svakom trenutku klijentima dostupna samo jedna kopija baze. Ova kopija je poznata kao glavna baza podataka, dok se sva ažuriranja koja su izvršili klijenti u glavnoj bazi primenjuju nad drugom kopijom baze, poznatom kao *mirror* baza podataka. *Mirroring* baza i primarna baza su čvrsto spregnute korišćenjem logova transakcija za svako umetanje, ažuriranje ili brisanje izvršeno u glavnoj bazi podataka. U slučaju bilo kakvog budućeg kvara, log transakcija je ključ za obnavljanje informacija iz jedne u drugu bazu, pri čemu u tom trenutku mirroring baza postaje glavna baza u sistemu. [1]

Ovaj rad će posebno govoriti o konceptima povezanim sa replikacijom podataka kao i o tome kako replikaciji može pridružiti tehnika *mirroring-*a sa ciljem pružanja rešenja za visoku dostupnost i oporavak od katastrofe u kontekstu *Microsoft Sql Server*[[1]](#footnote-1)baze podataka, (*MSSQL* u nastavku teksta), pri čemu će rad biti propraćen primerima korišćenjem relativno jednostavne demo baze.

# Replikacija u MSSQL-u

Replikacija u SQL Server-u predstavlja skup tehnologija za kopiranje i distribuciju podataka i objekata baze podataka iz jedne baze u drugu, a zatim sinhronizaciju između baza podataka radi održavanja konzistentnosti. Replikacija se koristi za distribuciju podataka na različitim geografskim lokacijama, sa ciljem olakšanog i bržeg pristupa od strane udaljenih i mobilnih korisnika putem različitih veza interneta.

Replikacija baze podataka može biti pojedinačna pojava ili trajni proces. Ona uključuje sve izvore podataka u distribuiranoj infrastrukturi organizacije. Distribuirani sistem upravljanja se koristi za kopiranje i pravilnu distribuciju podataka među svim izvorima. Sveukupno, distribuirani sistemi za upravljanje bazama podataka (DDBMS) rade na tome da osiguraju da se promene, umetanja i brisanja, koje su izvršene nad podacima na bilo kojoj datoj lokaciji, automatski odražavaju u podacima uskladištenim na svim ostalim lokacijama.

Klasični slučaj replikacije baze podataka uključuje jednu ili više aplikacija koje povezuju primarnu lokaciju za skladištenje sa sekundarnom lokacijom koja je često na drugoj lokaciji. Danas su te primarne i sekundarne lokacije za skladištenje najčešće pojedinačne izvorne baze podataka – kao što je SQL Server - kao i skladišta podataka koja objedinjuju podatke iz tih izvora, nudeći usluge skladištenja i analitike nad većim količinama podataka, pri čemu se često hostuju u oblaku.

Slika 1. Dijagram replikacije u SQL Serveru

## 2.1. Osnovni pojmovi i definicje u vezi sa replikacijom

U nastavku je dato objašnjenje osnovne terminologije koja je u vezi sa konceptom replikacije u SQL Serveru, pri čemu se koncept oslanja na sliku 1 iz prethodne tačke:

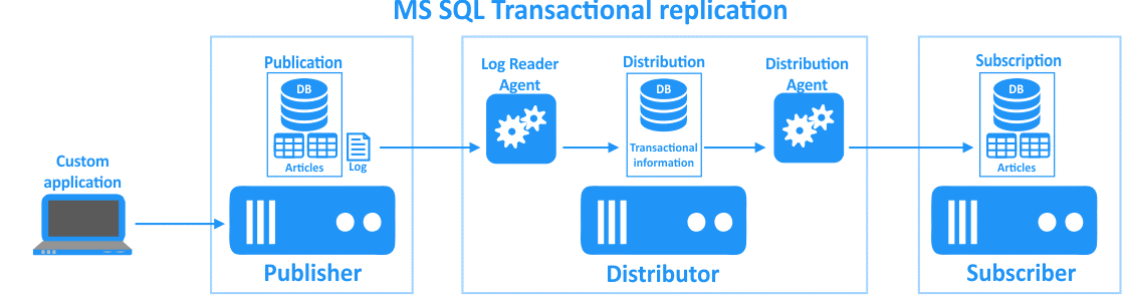
* Članak (engl. article) - Članak je osnovna jedinica SQL Server replikacije. Može se sastojati od tabela, uskladištenih procedura i pogleda. Pomoću njih se određuje koji objekti baze podataka će biti uključeni kao članak u publikaciji. Takođe ako se objavljuje objekat baze podataka koji zavisi od jednog ili više drugih objekata baze podataka, potrebno je objaviti sve referencirane objekte. Na primer, ako se objavljuje *view* koji zavisi od tabele, mora se objaviti i tabela. Sa druge strane, moguće je skalirati članak horizontalno i vertikalno korišćenjem filtara. Takođe možemo stvoriti više članaka nad istim objektom, uz neka ograničenja.
* Publikacija – Predstavlja logičku kolekciju članaka. Publikacija je zapravo finalni skup entiteta iz baze koji je određen za replikaciju.
* Filter – Predstavlja skup uslova definisan nad člankom. Ovaj koncept omogućava filtriranje, odnosno selekciju različitih entiteta za replikaciju pri čemu ima za posledicu smanjen saobraćaj na mreži kao i smanjenu redundansu odnosno količinu podataka koju čuva replika baze. Na ovaj način, može se vršiti replikacija samo ključnih i kritičnih podataka na replici.
* Izdavač (engl. publisher) – Izdavač odnosno glavna replika predstavlja server koji čuva glavnu kopiju baze podataka i služi za kreiranje i konfigurisanje publikacija. Ovaj server predstavlja bazu osnovnih dostupnih podataka za druge SQL Servere koji su konfigurisani da učestvuju u procesu replikacije neke organizacije.
* Pretplatnik (engl. subscriber) – Predstavlja bazu koja čuva replicirane podatke. Ovaj server prima podatke od izdavača, pri čemu je moguće konfigurisati različite oblike arhitekture *publisher-subscriber*. Moguće je konfigurisati *multi subscriber* model u kome postoji više pretplatnika konektovanih na jednu publikaciju ili postoji model samo jednog pretplatnika konektovanog na publikaciju. Takođe, jedan pretplatnik može primati podatke od starne više izdavača publikacija.
* Distributer - Distributer je takođe instanca MSSQL baze podataka, ali konfigurisana za prikupljanje transakcija iz publikacija i njihovu distribuciju pretplatnicima, kao što se može videti i sa dijagrama sa slike 1. Distributer se ponaša kao baza podataka za čuvanje repliciranih transakcija. Baza podataka distributera može se konfigurisati kao izdavač i distributer u jednom. U modelu lokalnog distributera, jedna instanca MS SQL Servera pokreće i izdavača i distributera. Model udaljenog distributera (situacija kada postoje zasebni serveri za izdavača i distributera) može se konfigurisati ukoliko je cilj organizacije da pretplatnici budu konfigurisani tako da koriste jednu MS SQL Server instancu za dobijanje različitih publikacija (pri čemu se ovaj model naziva i centralizovana distribucija).
* Agenti - Replikacija koristi brojne samostalne programe, koji se nazivaju agenti, i koji služe za izvršavanje zadataka vezanih za praćenje promena i distribuciju podataka. Po defaultu, agenti replikacije se izvode kao poslovi zakazani u okviru SQL Server Agent-a (SQL Server Agent hostuje i raspoređuje agente koji se koriste u replikaciji i pruža jednostavan način za pokretanje agenata replikacije). Agenti za replikaciju takođe se mogu pokretati iz komandne linije ili aplikacija koje koriste Objekte za upravljanje replikacijom (RMO). Agenti za replikaciju mogu se konfiigurisati i iz SQL Server Management Studija. U SQL Serveru postoji pet tipova agenata:
  + *Snapshot* agent – Ovaj agent se obično koristi u okviru svih vrsta replikacije. On priprema šeme i inicijalne datoteke publikovanih tabela i drugih objekata, zatim čuva *snapshot* datoteke i beleži informacije o sinhronizaciji u distributerskoj bazi podataka. Snapshot Agent se pokreće kao servis u okviru distributera.
  + Čitač logova – Ovaj agent se koristi u okviru transakcionih replikacija (o tipovima replikacija više u sledećem poglavlju). Njegov zadatak je da premesti one transakcije koje su označene za replikaciju iz transakcionog loga u distributerovu bazu. Pritom značajna je činjenica da će svaka baza u okviru transakcione replikacije imati zasebnog log agenta na distributeru, koji će biti konektovan na izdavačev server.
  + Distributer agent – Ovaj agent se koristi u sklopu transakcionih replikacija i *snapshot* replikacija. Njegov zadatak je da primeni inicijalni snapshot podataka u pretplatnikovoj bazi, i da zatim prebacuje transakcije sačuvane u distributerovoj bazi. U zavisnosti od tipa supskripcije - *push* ili *pull* – agent će biti pokrenut ili kod distributera ili kod pretplatnika.
  + *Merge* agent – kao što i samo ime govori, ovaj tip agenta biće pokrenut sa *merge* replikacijom. O će takođe da primeni prvo početni *snapshoot* u pretplatnikovoj bazi a zatim će vršiti usklađivanje inkrementalnih promena podataka koje se javljaju. Za svakog pretplatnika kreira se zasebni merge agent koji se povezuje i sa izdavačem i sa pretplatnikom i ažurira oba. Agent radi u okviru distributera za *push* pretplate ili u okviru pretplatnika za *pull* pretplate.
  + Čitač redova - Agent čitača redova koristi se sa replikacijom transakcija sa opcijom *queued updating*. Agent radi kod distributera i prenosi promene napravljene kod pretplatnika nazad kod izdavača. Za razliku od distributer agenta i agenta *merge* ovde postoji samo jedana instanca agenta čitača redova koji uslužuje sve izdavače i publikacije za datu bazu podataka.
* Supskripcija – Odnosno pretplata, predstavlja zahtev za kopijom publikacije koja se mora dostaviti pretplatniku. Pretplata se koristi za definisanje podataka publikacije koji se moraju primiti, kao i to gde i kada će se ti podaci biti primljeni. Postoje dve vrste pretplate - *push* and *pull*.
  + *Push* supskripcija - promenjeni podaci se *prisilno* prenose iz distributera u bazu podataka pretplatnika. Nije potreban pretplatnički zahtev.
  + *Pull* supskripcija - pretplatnik zahteva promenjene podatke napravljene u izdavaču. Agent radi na strani pretplatnika.
* Metapodaci – Predstavljaju podatke koji se koriste za opisivanje entiteta baze podataka. Postoji širok spektar ugrađenih funkcija metapodataka koje omogućavaju uzimanje informacija o instanci SQL Servera, instancama baze podataka i entitetima baze podataka.

## 2.2. Tipovi replikacije u SQL Serveru

U SQL Serveru postoje različiti tipovi replikacije, pri čemu njihov odabir može zavisiti od mnogo različitih faktora, uključujući fizičku arhitekturu koja je dostupna za replikaciju, zatim količinu podataka koja se koristi u ovom procesu, kao i sam model replikacije koji je izabran (na primer da li se podaci ažuriraju kod pretplatnika ili koliko pretplatnika je povezano sa koliko publikacija i slično).

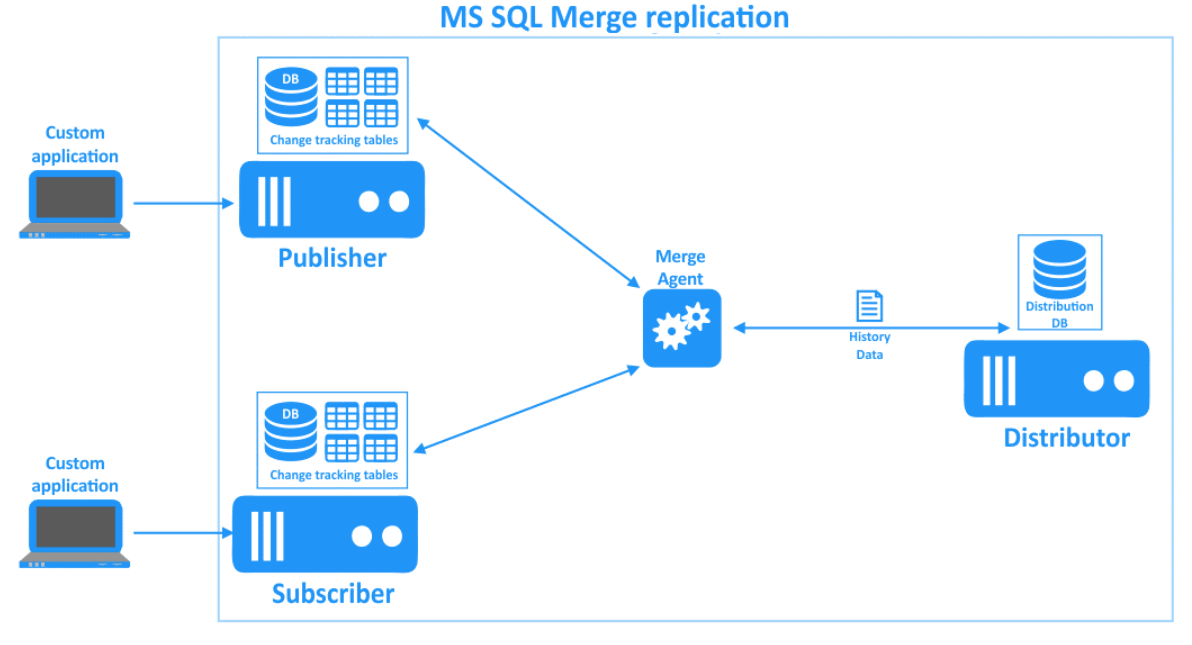
Generalno, svaki proces replikacije započinje početnom sinhronizacijom publikovanih objekata između izdavača i pretplatnika. Ova početna sinhronizacija se često izvršava korišćenjem *snapshot*-a, koji predstavlja kopiju svih objekata i podataka navedenih u publikaciji. Dok je za neke tipove aplikacija ovo dovoljno, za druge tipove važno je da naknadne, inkrementalne promene podataka tokom vremena postupno prelaze u bazu pretplatnika. Takođe postoje i aplikacije koje zahtevaju da promene pređu sa pretplatnika natrag u izdavača.

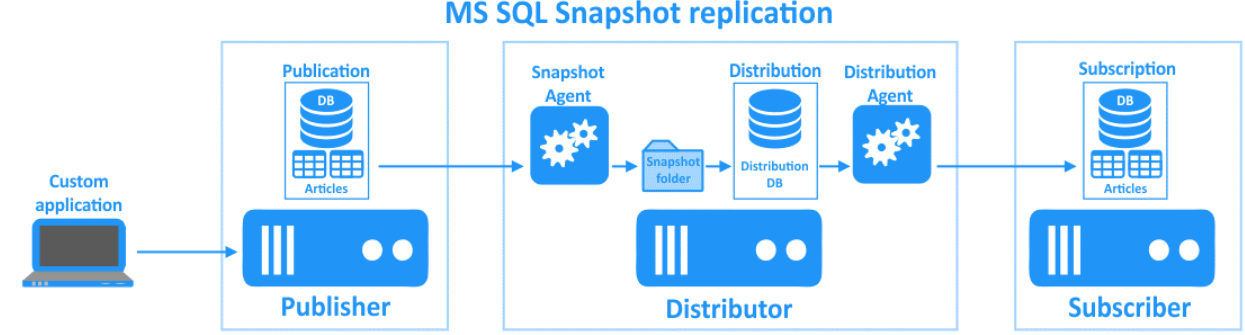
SQL Server nudi sledeće tipove replikacija:

* Transakcione replikacije - Transakciona replikacija je periodična automatizovana replikacija kod koje se podaci distribuiraju od glavne baze podataka do replike baze podataka u realnom vremenu (ili gotovo u realnom vremenu). Kod transakcionih replikacija ne samo da se replicira konačno stanje baze podataka, već se repliciraju i sve izvršene transakcije, što omogućava praćenje celokupne istorije transakcija i na replici baze podataka. Na početku procesa transakcione replikacije, kreirani *snapshot* se šalje pretplatniku, a zatim se podaci neprekidno prenose iz glavne baze podataka u repliku baze podataka nakon neke izvršene promene i to u istom redosledu u kome su se odigrale u glavnoj bazi, odnosno izdavaču. Transakciona replikacija se uglavnom koristi kao jednosmerna replikacija.

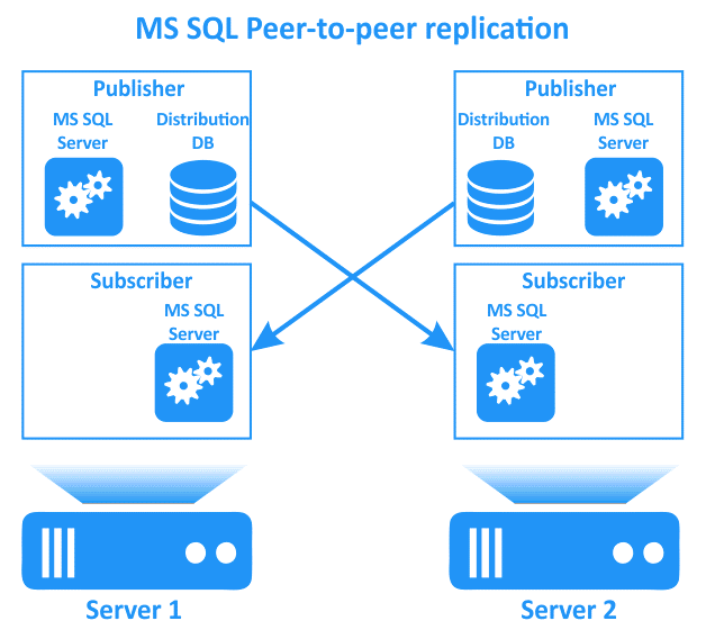
Slika 2.Dijagram transakcione replikacije u SQL Serveru

* *Merge* replikacija – Ova replikacija predstavlja vrstu dvosmerne replikacije koja se obično koristi u okruženjima *server-to-client,* za sinhronizaciju podataka na serverima, kada konekcija između njih nije stalna. Kada se mrežna veza uspostavi između oba servera baze podataka, *merge* agenti otkrivaju promene napravljene u obe baze i modifikuju baze podataka kako bi sinhronizovali i ažurirali njihovo stanje, pri čemu se promene koje se dešavaju i na izdavaču i na pretplatniku prate *trigger-*ima Ovaj tip replikacije sličan je transakcionoj replikaciji, pri čemu se ovde podaci repliciraju od izdavača do pretplatnika ali i obrnuto.

Slika 3. Merge replikacija u SQL Serveru

* *Snapshot* replikacija – Predstavlja jednostavniji tip replikacije pri čemu se sinhronizacija vrši tako što se primenjuje *snapshot* baze izdavača na bazu pretplatnika, ge izdavač koji distribuira podatke tačno onako kako se pojavljuju u određenom trenutku i ne nadgleda ažuriranja podataka. Kada dođe do sinhronizacije, ceo snimak se generiše i šalje pretplatnicima. Ova vrsta replikaciije korisna je u slučajevima kada je potrebno ažurirati podatke jednom dnevno iz baze glavnog servera u baze servera u poslovnicama.

Slika 4. Snapshot replikacija u SQL Serveru

* *Peer-to-peer* replikacija – Ova vrsta replikacije je napravljena po ugledu na transakcione replikacije, pri čemu prenosi transakciono dosledne promene u gotovo realnom vremenu između više instanci servera, odnosno pretplatnika. Ovaj tip replikacije je naročito zgodan za korišćenje u situaciji kada su serveri geografski distribuirani na više različitih lokacija. Promene se mogu izvršiti na bilo kom serveru baze podataka i šire se na sve servere baza podataka. Arhitektura koja dodatno objašnjava dati tip replikacije, prikazana je na sledećoj slici:

Slika 5.Peer to peer replikacija u SQL Serveru

* Bidirekciona replikacija – Predstavlja poseban tip transakcione replikacije kod koje je dozvoljeno da dva servera razmenjuju svoje promene u bazi podataka jedan sa drugim. Svaki server objavljuje podatke i prima podatke.
* Ažurirane pretplate – Predstavlja još jedan tip transakcionih replikacija kod kojih se podaci mogu ažurirati kod pretplatnika a zatim se šalju izdavaču koji potom propagira promene ostalim pretplatnicima.

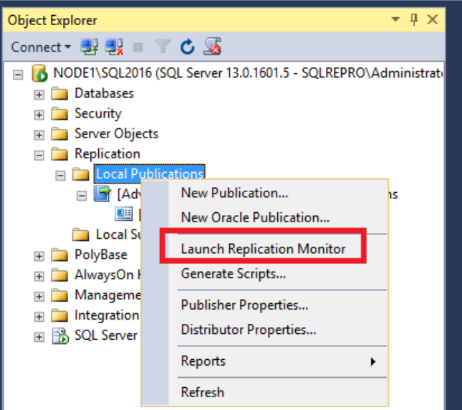
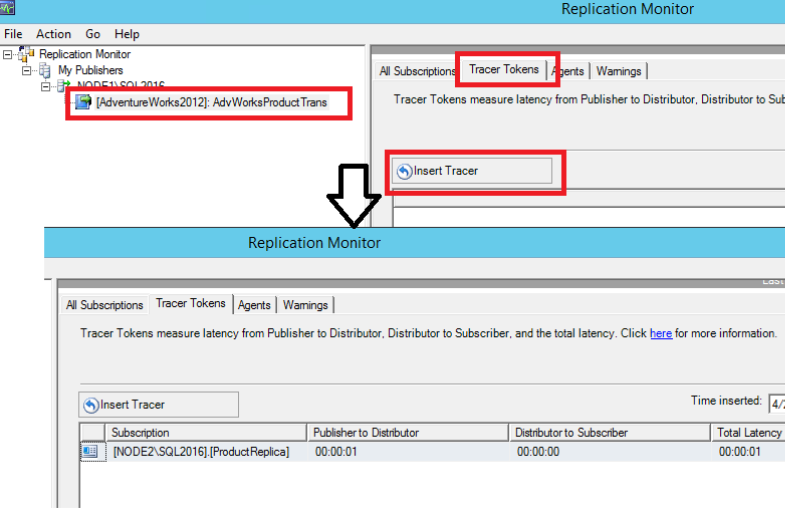
## 2.3. Primer koraka prilikom kreiranja komponenata transakcione replikacije u SQL Serveru

U nastvku ovog poglavlja biće dat primer koji ima za cilj ilustraciju rada procesa replikacije u okviru SQL Servera. Na samom početku potrebno je ispratiti korake za konfigurisanje uloga koje učestvuju u datom procesu, pri čemu će primer biti obrađen korišćenjem transakcione replikacije. Uloge koje je potrebno konfigurisati su sledeće:

* SQL Server distributer
* SQL Server izdavač
* SQL Server pretplatnik

Pritom, za konfiguraciju uloga distributera i izdavača moguće je iskoristiti jednu instancu SQL Servera, dok je za ulogu pretplatnika potrebno iskoristiti drugu instancu SQL Servera. U procesu konfiguracije uloga, potrebno je voditi računa o sledećim stvarima:

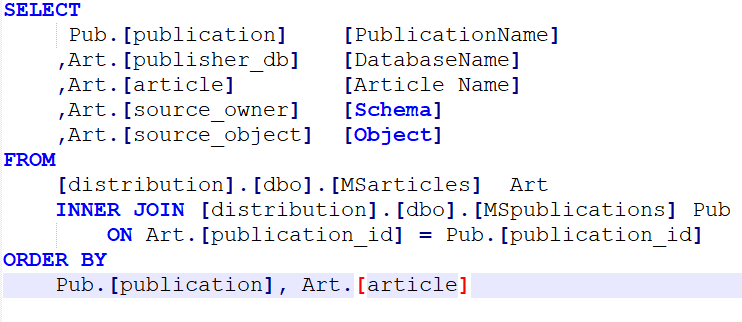
* Prilikom konfiguracije distributera neophodno je podesiti putanju za replikacioni *snapshot* direktorijum i dati lokaciju podataka i log fajlova.
* Prilikom konfigurisanja izdavača i kreiranja nove *publication* baze treba odabrati odgovarajuću bazu podataka koju je potrbno replicirati. Takođe u ovom koraku treba odabrati odgovarajući tip replikacije publikacije, pri čemu je ovde izabrana transakciona publikacija.
* U nastavku procesa konfiguracije izdavača neophodno je podesiti i članke (engl. articles). Objekti koji budu selektovani u ovom koraku će biti uključeni u okviru publikacije. Na raspolaganju za selekciju su:
  + Tabele
  + Sačuvane procedure
  + Pogledi
  + Indeksirani pogledi
  + Korisnički definisane funkcije
* U nastavku je moguće konfigurisati i potrebne filtre nad zapisima u tabelama koji će biti primenjeni nad selektovanim člancima.
* Nakon toga se vrši podešavanje startovanja *snapshot* agenta, pri čemu se on može pokrenuti ili odmah ili se može podesiti njegovo naknadno vreme izvršenja.
* Na kraju potrebno je kreirari i pretplatnika. U okviru ovog *setup*-a moguće je kreirati *push* odnosno *pull* pretplatnika izborom odgovrajuće opcije, pri čemu je u konkretnom primeru kreiran *push* pretplatnik. Vrsta pretplatnika naravno zavisi od toga na kojoj strani će agenti biti pokrenuti – ili u okiviru distributera ili u okviru pretplatnika. Za pretplatnika je takođe potreno kreirati novu bazu koja će predtavljati replika bazu podataka.

Nakon kompletiranja konfiguracije, može se vršiti nadgledanje statusa agenata i replikacije pokretanjem *Replication Monitor*-a u okviru *SQL Server Management Studio*-a.

Slika 6. Iustraacija pokretanja Replication Monitora korišćenjem tracer tokena za praćenje promena u bazi pretplatnika [5]

Na ovaj način, korišćenjem *tracer* tokena moguće je izvršiti praćenje sa ciljem verifikacije promena u bazi pretplatnika i sa ciljem utvrđivanja kašnjenja. Latenca je vreme potrebno da se pretplatniku prikaže promena izvršena kod izdavača.

## 2.4. Primeri kreiranja i pregleda članaka kod transakcione replikacije SQL Servera

Kao što je već rečeno, članci predstavljaju osnovu SQL Server publikacije. Oni se sajstoje od tabela, sačuvanih procedura i pogleda. Za nadgledanje svih objavljenih članaka, može se izvršiti sledeća T-SQL naredba

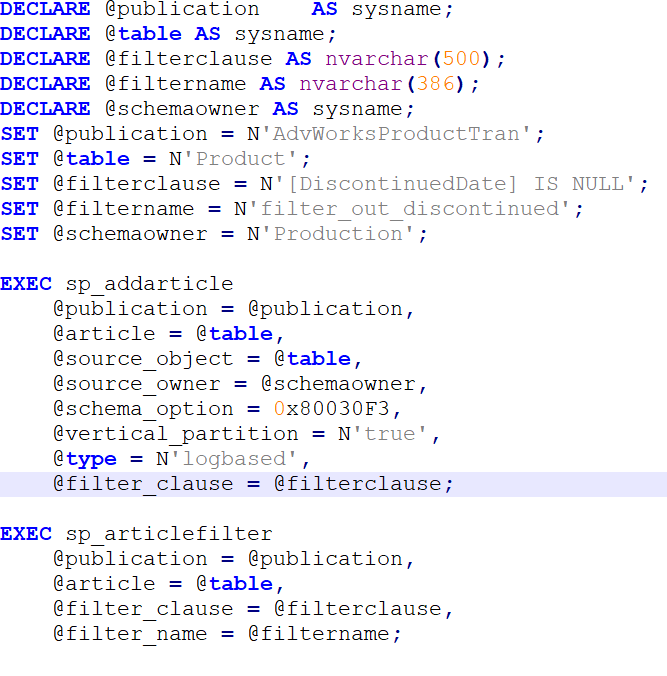
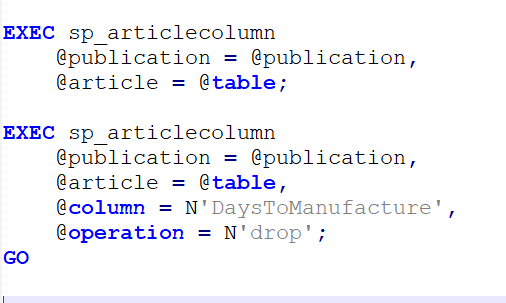
Slika 7. Listanje svih objavljenih članaka

Za definisanje novog članka potrebno je izvršiti sledeće:

* Na strani izdavača potrebno je izvršiti naredbu *sp\_addarticle* ( naredba predstavlja sačuvanu proceduru koja vrši kreiranje članka i njegovo dodavanje u publikaciju).
* Zatim potrebno je specificirati ime publikacije kojoj dati članak pripada korišćenjem *@publication*, ime koje nosi sam članak *@article,* izvorni objekat koji će biti objavljen *@source\_object,* kao i tip *@type* ukoliko članak nije *log-based* tabela.
* Ukoliko je potrebno dodati filtriranje moguće su sledeće opcije:
  + Za definisanje horizontalnih filtara (filtriranje zapisa u tabeli ili pogledu članka), potrebno je koristiti *sp\_articlefilter* sačuvanu proceduru u okviru izdavača.
  + Za vertikalno filtriranje (filtriranje kolona u tabeli ili pogledu članka) potrebno je koristiti *sp\_articlecolumn* sačuvanu proceduru na strani izdavača.
* Ukoliko je članak filtriran potrebno je izvršiti i *sp\_articleview* proceduru koja stvara pogled koji definiše objavljeni članak kada se tabela filtrira vertikalno ili horizontalno. Ovaj pogled se koristi kao filtrirani izvor šeme i podataka za odredišne tabele.
* Ukoliko publikacija već poseduje postojeće pretplate, i funkcija *sp\_helppublication* vrati vrednost nula za kolonu *immediate\_sync* onda je neophodno pozvati funkciju *sp\_addsubscription* kako bi čanak bio dodat u svaku postojeću supskripciju.
* Na kraju, ukoliko publikacija poseduje *pull* pretplate, potrebno je pozvati i funkciju *sp\_refreshsubscriptions* na strani izdavača kako bi se kreirao novi *snapshot* za postojeće *pull* supskripcije koji sadrži samo novi članak.

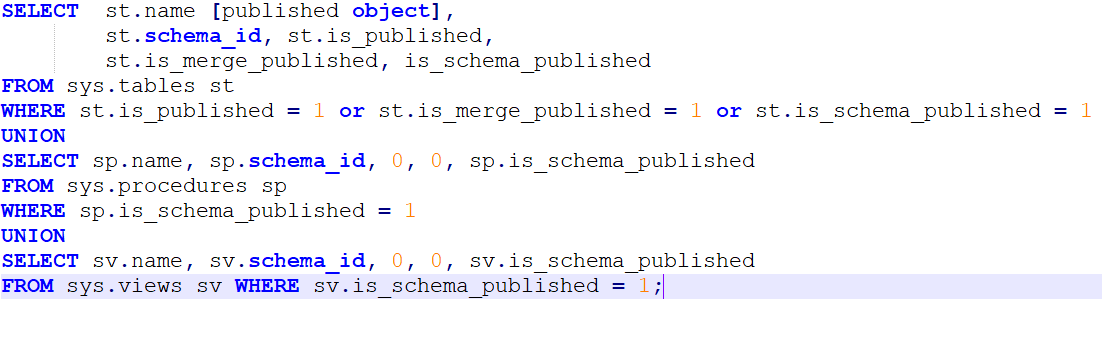
Primer prethodno opisanog procesa dodavanja novog članka dat je na sledećim slikama. Ovaj primer definiše članak nad tabelom Proizvod, a pritom vrši vertikalno i horizontalno filtriranje članka.

Slika 8 prikazuje prvi deo naredbi za dodavanje novog članka. Na početku se vrši setovanje imena publikacije, filtara (horizontalnih i vertikalnih), imena članka, izvornog objekta i vlasnika šeme. Šema opcija *@schema\_option* predstavlja bit masku za opcije generisanja šeme za dati članak. Ona služi za kontrolu opcije kreiranja objekata koji se repliciraju za objavljeni objekat. Ovde je vrednost šema opcije postavljena na 0x8000000, čime se specificira kreiranje svih šema koje nisu već prisutne na strani pretplatnika.

Slika 8. Kreiranje članka transakcione replikacije – prvi deo

Slika 9. Kreiranje članka transakcione replikacije – drugi deo

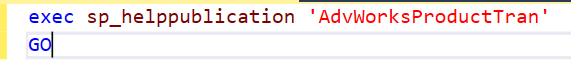
Na slici 9, prva *exec* naredba (*exec* je SQL naredba za izvršenje sačuvanih procedura) vrši dodavanje svih kolona u članak, dok naredna *exec* naredba vrši vertikalno filtriranje uklanjanjem kolone *DaysToManufacture* iz članka.

Dalje, za preuzimanje informacija o nekom članku transakcione ili *merge* replikacije u publikovanoj bazi potrebno je izvršiti sledeću TSQL naredbu:

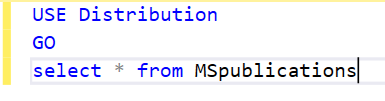
Slika 10. Preuzimanje informacija o članku

Kao što se vidi na slici 10, za preuzimanje informacija o nekom članku koriste se informacije iz sistemske tabele, sistemskog pogleda i sistemskih procedura (ove tabele predstavljaju sistemske tabele koje se koriste za održavanje informacija o svim tabelama odnosno pogledima odnosno sačuvanim procedurama u bazi podataka).

## 2.5. Primeri preuzimanja informacija o ostalim komponentama replikacije

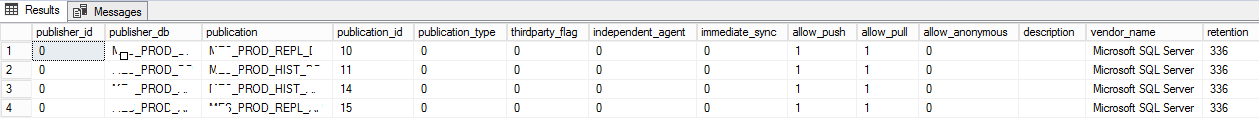
* Publikacija predstavlja logičku kolekciju članaka iz baze. Za pregled publikacije koristi se sačuvana procedura *sp\_helppublication* koja vraća informacije o publikaciji. Ova procedura se izvršava na strani izdavača.

Slika 11. Preuzimanje informacija o publikaciji

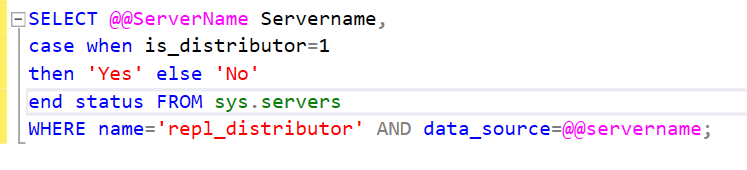
* Za pregled baze publikacija potrebno je izvršiti sledeću naredbu:

Slika 12. Preuzimanje informacija iz baze izdavača

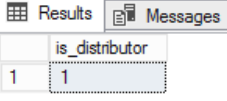
Bitno je naglasiti da izdavač može imati jednu ili više publikacija. Pritom svaki izdavač definiše mehanizam propagacije podataka stvaranjem nekoliko internih uskladištenih procedura.

Rezultat nakon poziva prethodne naredbe:

Slika 13. Rezultat nakon preuzimanja informacija iz baze izdavača

* Još jedna korisna naredba za preuzimanje dodatnih informacija o replikaciji predstavlja sledeću naredbu koja ima za cilj da utvrdi da li je server distributer ili ne:

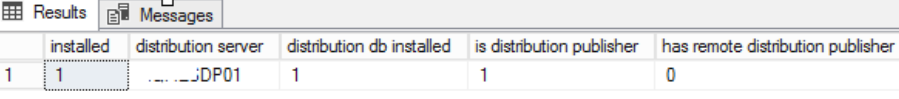
Slika 14. Provera da li je server distributer

Rezultat nakon izvršenja prethodne naredbe:

Slika 15. Rezultat naredbe sa slike 14

* Takođe koriščenjem procedure *sp\_get\_distributor* može se utvrditi da li je distributer instaliran na serveru. Ova naredba se izvršava na računaru na kojem se traži distributer, u bilo kojoj bazi podataka.

Slika 16. Provera distributera



Slika 17. Rezultat naredbe sa slike 16

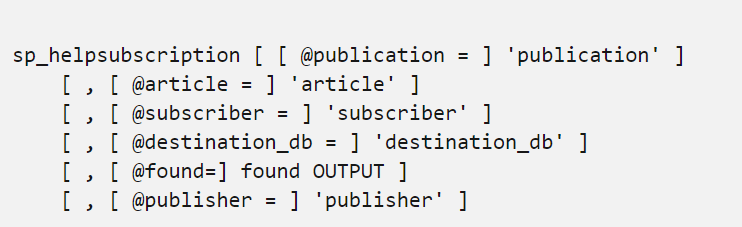
* Instanca baze podataka koja uzima SQL Server replikacione podatke iz publikacije je, kao što je prethodno pomenuto, zadatak pretplatnika. Za prikaz informacija o pretplatniku potrebno je pozvati sledeću uskladištenu proceduru na strani izdavača:

Slika 18. Preuzimanje informacija o pretplatniku

* Za preuzimanje informacija o određenoj supskripciji, koja zapravo predstavlja zahtev za kopijom određene publikacije, može se iskoristiti *sql stored* procedura *sp\_helpsubscription.* Ona služi da izlista sve informacije povezane sa određenom publikacijom, člankom, Pretplatnikom ili skupom supskripcija. Ova procedura se izvršava na strani Izdavača u bazi publikacija.



Slika 19. Preuzimanje informacija o supskripcijama

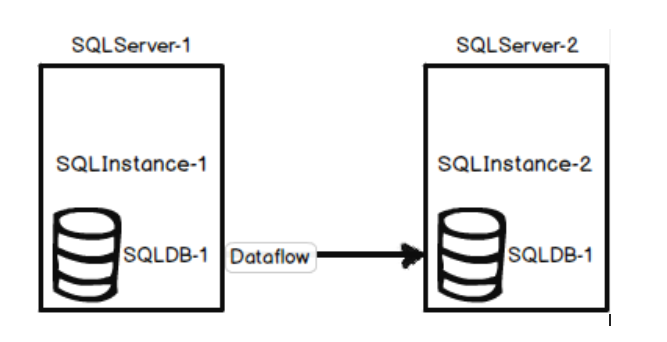
Za većinu prethodno pomenutih sačuvanih procedura SQL Servera postoji dodatni broj argumenata koje je moguće navesti sa ciljem specificiranja konkretnijih informacija u vezi sa datom replikacijom. Za slučaj funkcije iz posledenje stavke, slede argumenti koje je moguće navesti u okviru njenog poziva:

Slika 20. Funkcija sp\_helpsubscription sa argumentima [7]

# *Mirroring* u SQL Serveru

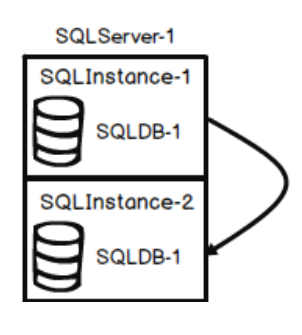
*Mirroring* baze podataka, predstavlja tehniku SQL Servera za oporavak podataka u slučaju katastrofe pri čemu se može koristiti zajedno sa replikacijom kako bi se poboljšala dostupnost baze podataka publikacije. Ova tehnika uključuje dve instance SQL Servera na, uglavnom, različitim mašinama. Jedna instanca SQL Servera deluje kao primarna instanca koja se naziva primarna baza, dok je druga *mirror* instanca i zove se ogledalo baza.

U opštem slučaju možemo razlikovati sledeće dve konfiguracije mirroring tehnike:

* Konfiguracija predstavlja okruženje sa dva SQL Servera pri čemu postoje dve instance *SQLInstance-1* i *SQLInstance*-*2* baze i jedna *mirrored* baza*:*

Slika 21. Primer najčešće konfiguracije Mirroring baze

* Konfiguracija predstavljena okruženjem sa jednom SQL mašinom, ali sa dve SQL Server instance na njoj, uključujući opet i *mirrored* bazu *SQLDB-1.*

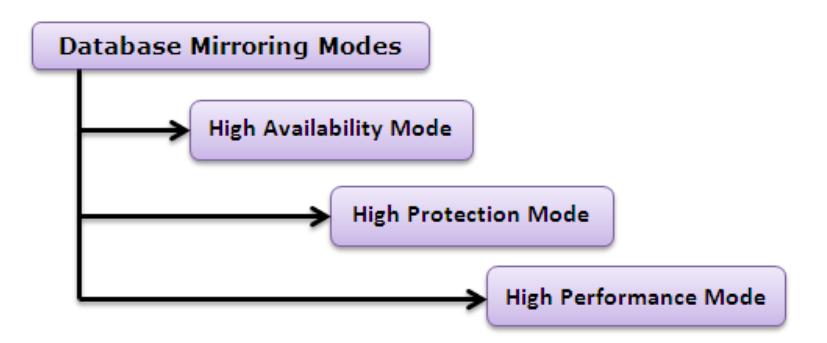
Slika 22. Primer konfiguracije na jednoj mašini

* Takođe postoji mogućnost dodavanja i treće, opcione komponente, odnosno servera zvanog svedok server (engl. witness). Kada je prisutan, ovaj server nadgleda glavni i *mirror*-ovani server sa ciljem da osigura komunikaciju i konekciju između komponenti koje učestvuju u *mirror* sesiji, koristeći kvorum. Ukoliko bilo koji od servera izgubi kvorum, svedok server dodeljuje ulogu glavnog servera, uzrokujući automatsko preusmeravanje glavnog na mirrorovani server, ukoliko je to neophodno. Za mod rada u automatskom preusmeravanju (više u sledećem poglavlju) neophodno je koristiti svedoka.

Prilikom konfigurisanja mirrored baze potrebno je sa druge strane imati na umu i sledeće zahteve i ograničenja:

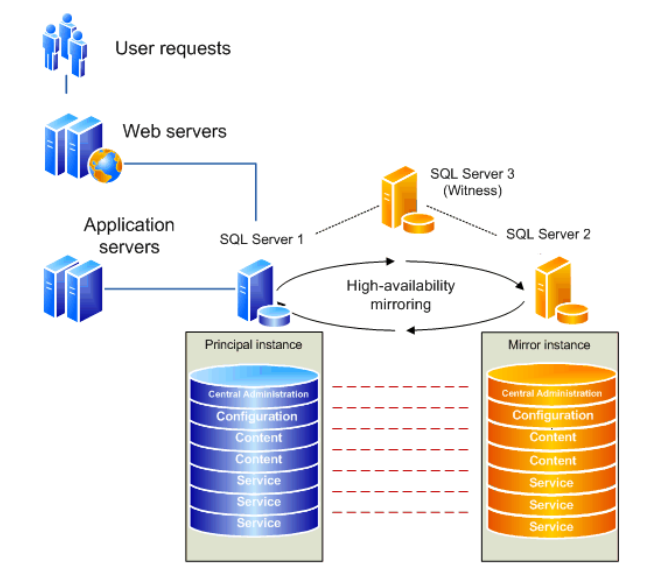
* Glavna baza i *mirrored* baza moraju da dele distributera, pri čemu je najbolje konfigurisati arhitekturu tako da je distributer na udaljenom serveru.
* U SQL Serveru replikacija podržava *mirroring* tehniku samo u slučaju transakcionih i *merge* replikacija i to u sklopu sa *read-only* pretplatnicima ili pretplatnicima koji se ažuriraju koristeći redove.
* Mirroring baza neće sadržati sve podatke, poput metapodataka ili objekata koji postoje izvan same baze podataka (loginovi, povezani serveri itd.).

## 3.1. Operativni modovi *mirroring* baze

U SQL Serveru, mirror baza može biti podešena tako da pruža visoku dostupnost podataka ili oporavak od katastrofe.

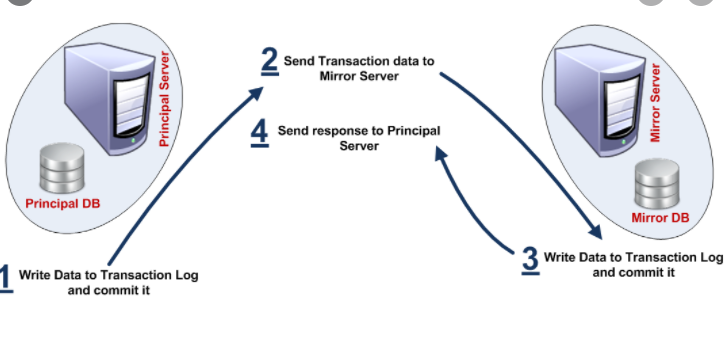
Slika 23. Modovi rada mirroring-a

U zavisnosti od potreba aplikacije, može se izabrati jedan od sledeća tri moda rada:

* Mod visoke sigurnosti – U modu visoke sigurnosti podaci moraju biti sinhronizovani, odnosno *commitovani* i na glavnoj i na *mirrorovanoj* bazi. Tek nakon izvršenja ove sinhronizacije, aplikacija može nastaviti dalje sa normalnim radom. Jasno je da će ovaj način funkcionisanja i sinhronizacije između ove dve baze dovesti do pogoršanja performansi, odnosno posledica je da će doći do dužeg čekanja pre nego što aplikaciji bude dozvoljeno da nastavi sa daljim radom. U slučaju pada glavne baze, postoje dve mogućnosti delovanja:
  + Ne raditi ništa – čekati na glavnu bazu da opet postane dostupna. Nakon toga proces mirroring-a se nastavlja tamo gde je zaustavljen.
  + Pokrenuti SQL Server instancu na *mirror-*ovanoj bazi – u ovom trenutku *mirror* baza postaje glavna baza podataka. U toku ovog procesa može doći do gubitka nekih podataka – onih koji još uvek nisu sačuvani na *mirror* bazi u trenutku kada je pala glavna baza (što se neće desiti ukoliko se primeni scenario iz prethodne tačke).
*  Mod visoke sigurnosti sa automatskim preusmeravanjem u slučaju otkaza – U ovom operativnom modu potrebno je koristiti tri servera, odnosno potrebno je koristiti i servera svedoka. Podatke je ponovo potrebno sinhronizovati izmedju glavne i *mirror* baze. Ukoliko sada dođe do otkaza automatski će doći do pokretanja preusmeravanja glavne na *mirror* bazu, bez mogućnosti odabira kao u prethodnom modu.

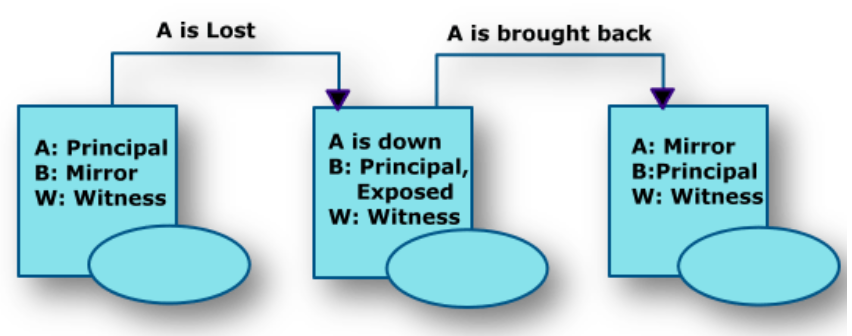
Slika 24. Opeativni mod visoke sigurnosti sa automatskim preusmeravanjem

* Mod visokih performansi – Ovaj mod je dostupan samo u *Enterprise edition* SQL Serveru. U ovom modu koristi se asinhrona komunikacija, pri čemu se podaci *commit*-uju na glavnom serveru, a onda kasnije šalju na *mirror*-ovani server. Sada ukoliko glavna baza postane nedostupna, postoje tri opcije:
  + Ne raditi ništa – čekati na glavnu bazu da ponovo postane dostupna.
  + Pokrenuti SQL Server instancu na *mirror-*ovanoj bazi tako da ona sada postane glavna baza.
  + Ručno ažuriranje – ukoliko pali server to dozvoli, moguće je uzeti krajnji deo bekap loga i obnoviti ga na *mirror* bazi (uz prethodno uklanjanje *mirror-*ovanja). Primer rada u modu ručnog ažuriranja prikazan je na sledećoj slici:

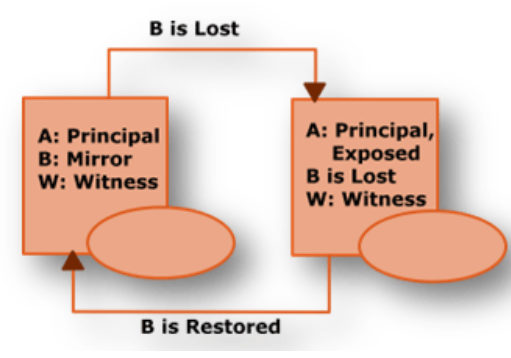
Slika 24. Ručno ažuriranje u modu visokih performansi

### Otkaz servera u modu visoke sigurnosti sa automatskim preusmeravanjem

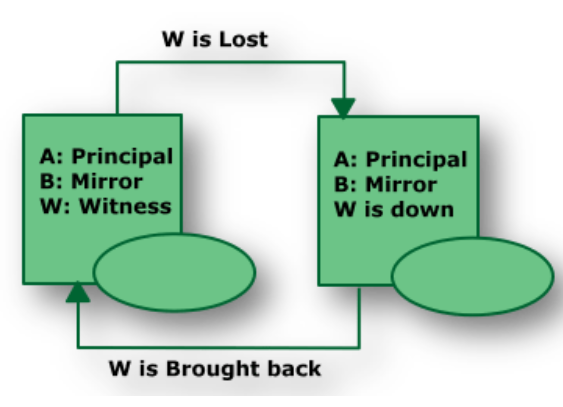
U modu visoke sigurnosti sa automatskim preusmeravanjem, postoje sledeće tri mogućnosti otkaza:

* Otkaz glavnog servera – Na sledećoj slici prikazana je situacija u slučaju otkaza glavnog servera. Ukoliko glavni server otkaže, *mirror* server preuzima ubrzo njegovu ulogu. Nakon oporavka bivšeg glavnog servera, on postaje novi mirror server. Uloga svedoka je da nadgleda i pokreće ovaj proces.

Slika 25. Otkaz glavnog servera

* Otkaz *mirror* servera – Gubitak *mirror* servera je situacija kada ne dolazi ni do kakve promene u ulogama servera. Ovo predstavlja potencijalno opasnu situaciju s obzirom da je sada glavni server izložen opasnosti jer ne može slati podatke na *mirror* server.

Slika 26. Otkaz mirror servera

* Otkaz svedok servera – U ovoj situaciji proces *mirroring*-a se nastavlja, ali operativni mod automatskog preusmeravanja više nije moguć.

Slika 27. Otkaz witness servera

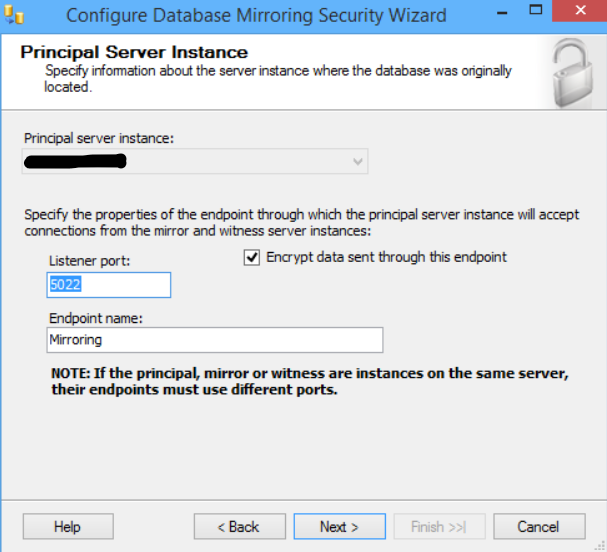
## Podešavanje database mirroring okruženja u SQL Serveru

Za podešavanje replikacije i *mirroring*-a potrebno je ispratiti sledeće korake prema uputstvu koje pruža oficijalna dokumentacija [9]:

* Konfigurisanje Izdavača
* Kofigurisanje *mirroring*-a
* Konfigurisanje *mirror* servera tako da može da koristi istog Distributera
* Konfigurisanje replikacionih agenata za preusmeravanje
* Dodavanje glavnog i *mirror* servera u replikacioni monitor

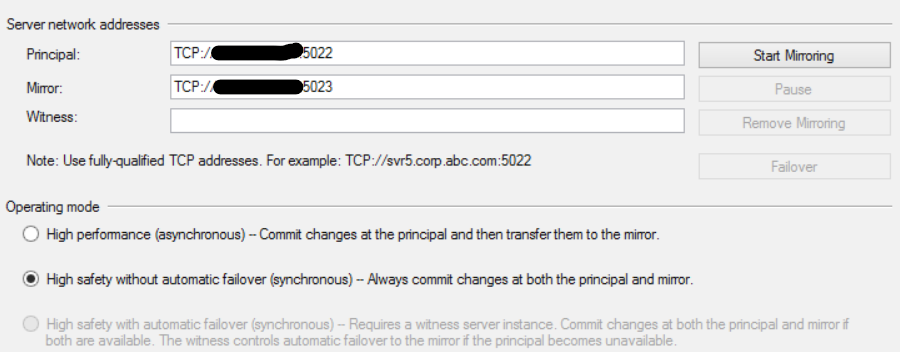
Na početku je neophodno postaviti bazu podataka koja će biti *mirror*-ovana u *full recovery* model. Takođe bitno je zapaziti da sistemske baze ne mogu biti *mirror-*ovane. Zatim je potrebno kompletnu bazu podataka zajedno sa transakcionim logovima moraju biti restorovani na SQL Server instanci koja će imati ulogu *mirror*-a (pri čemu *restore* proces mora biti izvršen koristeći NO RECOVERY opciju).

Podešavanje *mirroring*-a je potrebno na početku inicirati sa glavnog servera koristeći *SQL Server Management Studio wizard* ili TSQL naredbe. Tokom ovok procesa potrebno je obratiti pažnju na sledeća podešavanja:

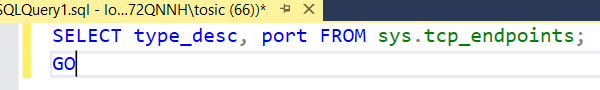
* Odabir *witness* servera – ova opcija je jedino neophodna ukoliko je potrebno podesiti operativni mod *mirrror* baze da bude mod visoke sigurnosti sa automatskim preusmeravanjem.
* Podešavanje komunikacije odnosno krajnjih tačaka – podešavanje porta i imena servera (podešavanja je potrebno uraditi i na *mirror* i na glavnom serveru)

Slika 28. Konfigurisanje mirror server endpoint-a

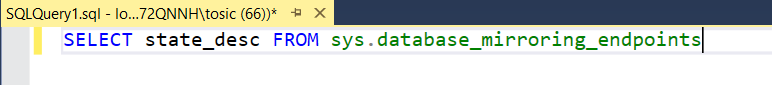
* Nakon ovoga potrebno je podesiti i operativni mod – mod se bira između *high safety* ili *high performance*. Ukoliko je prethodno bio specificiran svedok server, odabir *high safety* moda će rezultovati u odabiru moda visoke sigurnosti sa automatskim preusmeravanjem.

Slika 29. Podešavanje operativnog moda mirror servera

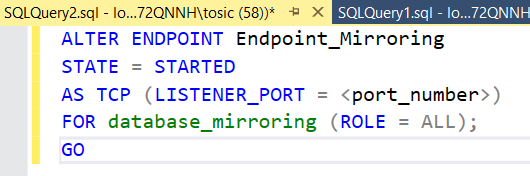
### Provera konfiguracije *endpoint*-a

Krajne tačke (engl. endpoint) moraju biti korektno konfigurisane, svaka instanca servera (glavni, mirrorovani ili svedok sevrer) moraju imati konfigurisan *mirroring endpoint*, kao što je prethodno prikazano. Za proveru portova na svakoj od SQL Server instanci, može se iskoristiti sledeća T-SQL naredba:

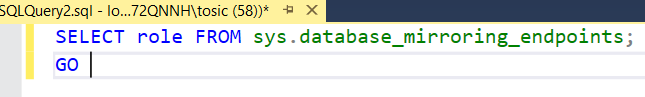
Slika 30. Provarea endpointa

Takođe, moguće je izvršiti proveru da li su odgovarajuće instance servera startovane:

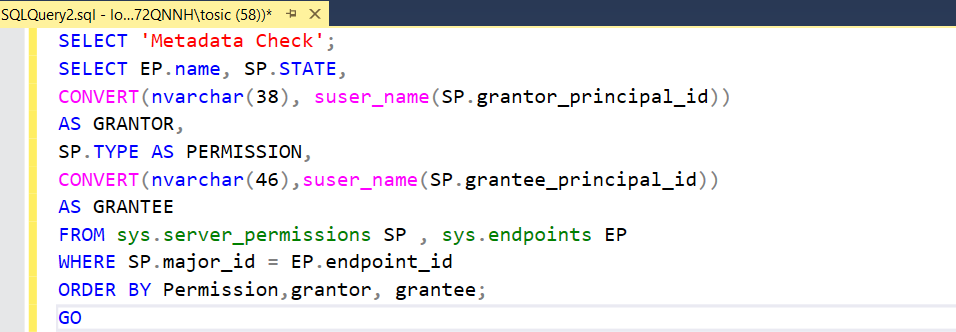
Slika 31. Provera stanja (STATE=STARTED) instanci servera

Ukoliko instance nisu startovane, pokretanje krajnjih tačaka je moguće izvršiti korišćenjem sledeće T-SQL naredbe:

Slika 32. Pokretanje endpointa

Takođe moguće je izvršiti proveru uloga, S tim ciljem na svakoj SQL Sevrer instanci moguće je pokrenuti sledeću T-SQL naredbu:

Slika 33. Provera uloga

Na kraju može se izvršiti provera *connect* dozvole. Kako bi se ustanovilo ko ima ovu dozvolu na svakoj instanci servera može se izvršiti provera korišćenjem sledećih naredbi:

Slika 34. Provera dozvole connect-a

## Razlike između replikacije i mirroring-a

Na kraju, možemo zaključiti razlike između replikacije i *mirroring*-a. U tu svrhu potrebno je ponovo ustanoviti jasne definicije ovih tehnika

* *Mirroring* – Proces kreiranja kopije podataka odnosno baze podataka. Generalno bi *mirroring* trebalo da kopira podatke na skroz drugu mašinu odnosno lokaciju koja je drugačija u odnosu na primarni server, kako bi ukoliko dođe do pada glavnog servera, *mirroring* instanca mogla da preuzme ulogu novog glavnog servera. Takođe bitno je uočiti da se mirroring vrši nad bazom i da ova tehnika ne podržava distribuirane baze podataka.
* Replikacija – Predstavlja način distribucije objekata baze podataka ka različitim bazama, sa ciljem poboljšanja procesa snadbevanja informacija. Glavna karakteristika tehnike replikacije jeste njena sposobnost da distribuira poodatke ka različitim, geografski udaljenim serverima. Na ovaj način se povećava paralelizacija. Jedna od glavnih komponenta procesa je izdavač koji poseduje informacije o replikaciji za različite servere.

Na osnovu prethodnih definicija moguće je zaključiti sledeće razlike koje postoje između datih tehnika SQL Servera:

* Dok se replikacija sprovodi nad objektima baze, odnosno podacima, dok se mirroring sprovodi nad celom bazom podataka
* U replikaciji nema rezervnih servera – svi serveri deluju kao aktivni serveri
* *Mirroring* je skuplja operacija u odnosu na replikaciju
* *Mirroring* ne podržava distribuirane baze podataka, dok replikacija podržava
* Sa gledišta lokacije, *mirroring* zahteva drugu mašinu, dok se kod replikacije očekuje samo druga baza.
* Glavna prednost replikacije je ta što povećava dostupnost i skalabilnost podataka, dok je kod *mirroringa* glavna prednost ta što smanjuje mogućnost da dođe do planiranih ili neplaniranih situacija kada glavni server nije dostupan.

# Zaključak

Korišćenje *SQL Server database mirroring* tehnike poseduje puno prednosti, poput činjenice da predstavlja ugrađeni SQL Server fičer, koji je relativno jednostavan za podešavanje i koji pruža automatsko preusmeravanje u slučaju otkaza u modu visoke sigurnosti, kao i još mnogo toga. Takođe, *database mirroring* može biti kombinovan sa drugim tehnikama za oporavak od otkaza, poput klasterizacije, *log shipping*-a i na kraju naravno replikacije.

Sa druge strane *mirroring* će verovatno biti uklonjen iz SQL Servera u budućim verzijama u korist *AlwaysOn Availability* grupa. Takođe mirroring je *per database only* rešenje što znači da *login*-ovi i poslovi iz glavne SQL Server baze moraju biti ručno rekreirani na *mirror* bazi. Uz to, može doći i do kašnjenja koje sa druge strane može biti sprečeno uglavnom samo korišćenjem boljeg i jačeg hardvera.

Na kraju, iako između ovih tehnika postoje neke bitne razlike treba imati na umu da se tehnike replikacije i *mirroring-*a koriste u kombinaciji kako bi se postigla veća dostupnost baza podataka i to najčešće za bazu publikacije, što u mnogim biznis aplikacijama može predstavljati neophodnu osobinu.

# Literatura

[1] Mirroring i replikacija u SSQL Serveru: <https://docs.microsoft.com/en-us/sql/database-engine/database-mirroring/database-mirroring-and-replication-sql-server?view=sql-server-ver15>

[2] Replikacija u SQL Serveru: <https://docs.microsoft.com/en-us/sql/relational-databases/replication/sql-server-replication?view=sql-server-ver15>

[3] Agenti replikacije u SQL Serveru: <https://docs.microsoft.com/en-us/sql/relational-databases/replication/agents/replication-agents-overview?view=sql-server-ver15>

[4] Konfigurisanje SQLServer replikacije <https://www.spotlightcloud.io/blog/setting-up-and-configuring-sql-server-replication>

[5] Konfigurisanje transakcione replikacije između dva potpuno povezana servera <https://docs.microsoft.com/en-us/sql/relational-databases/replication/tutorial-replicating-data-between-continuously-connected-servers?view=sql-server-ver15>

[6] Sačuvane procedure u SQL Serveru <https://docs.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/replication-stored-procedures-transact-sql?view=sql-server-ver15>

[7] Funkcija *sp\_helpsubscription* u SQL Serveru <https://docs.microsoft.com/en-us/sql/relational-databases/system-stored-procedures/sp-helpsubscription-transact-sql?view=sql-server-ver15>

[8] Replikacija i mirroring u SQL Serveru <https://docs.microsoft.com/en-us/sql/database-engine/database-mirroring/database-mirroring-and-replication-sql-server?view=sql-server-ver15>

[9] Mirroring i replikacija u SLQ Serveru <https://docs.microsoft.com/en-us/sql/database-engine/database-mirroring/database-mirroring-and-replication-sql-server?view=sql-server-ver15>

[10] Razlika između *mirroringa* i replikacije <https://techdifferences.com/difference-between-mirroring-and-replication.html>

[11] Razlika između *mirroringa* i replikacije <https://www.geeksforgeeks.org/difference-between-mirroring-and-replication/>

1. U trenutku pisanja ovog rada najnovija verzija SQL Servera je Microsoft SQL Server 2019, RTM 15.0.2000.5. [↑](#footnote-ref-1)