OBNOVA BAZE PODATAKA

IX predavanje

Dr.sc. Emir Mešković

Obnova baze podataka u slučaju razrušenja – Database Recovery

- Dovesti bazu podataka u najnovije stanje za koje se pouzdano zna da je bilo ispravno
- Velike baze podataka – dijeljene, višekorisničke – obavezno moraju posjedovati mehanizme obnove
- Male, jednokorisničke baze podataka obično imaju malu ili uopšte nemaju podršku obnovi – obnova se prepušta korisnikovoj odgovornosti
  - podrazumijeva se da korisnik periodično stvara backup kopiju te u slučaju potrebe obnavlja bazu ručno.

Uzroci razrušenja

- greške opreme
- greške operativnog sistema
- greške sistema za upravljanje bazama podataka
- greške aplikacijskog programa
- greške operatera
- kolebanje izvora energije
- požar, sabotaža, ...

Opšte pravilo koje omogućava obnovu

- Redundancija - svaki se podatak mora moći rekonstruisati iz nekih drugih informacija redundantno pohranjenih negdje drugo u sistemu (na traci, na drugom disku, na "ogledalnom" disku, ...)
- Redundancija se postiže:
  - "ogledanjem" podataka (*mirroring*)
  - sigurnosnim kopijama (*backup*)
  - dnevnicima izmjena (*logical log*) koji služe za:
    - poništavanje transakcija
    - ponovo obavljanje transakcija

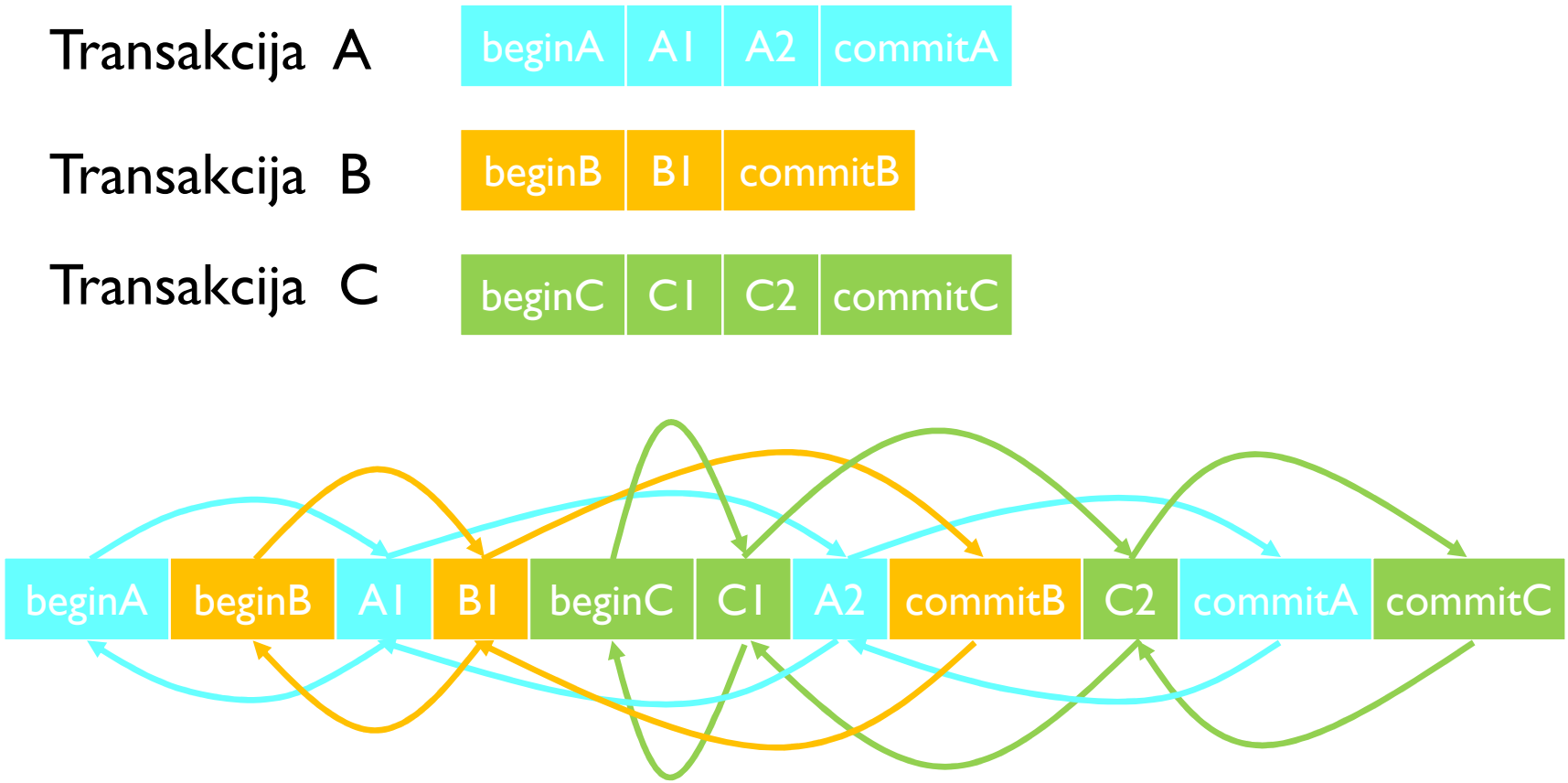
Opšti opis postupka koji omogućava obnovu

- Periodično kopiranje sadržaja baze podataka na arhivski medij (drugi disk, diskovni automat (*jukebox*) specijaliziranih sistema za sigurnosne kopije; nekad su to bile magnetske trake)
  - (1 × dnevno, 1 × sedmično - zavisno o učestalosti promjena)
- Svaka izmjena u bazi podataka evidentira se u logičkom dnevniku izmjena (*logical log*, *journal*)
  - stara vrijednost zapisa
  - nova vrijednost zapisa
  - korisnik, vrijeme, …
  - izmjena se prvo zapisuje u dnevnik, a tek se onda provodi!

Kad nastane kvar ...

- Baza je potpuno uništena
  - Učitava se najsvježija arhivska kopija (naredbom ROLLFORWARD DATABASE) – time se baza podataka dovodi u stanje kakvo je bilo u trenutku kad je napravljena posljednja arhivska kopija
  - Koristeći dnevnik izmjena ponovo se obavljaju izmjene koje su se dogodile u međuvremenu (nakon izrade arhive)
- Baza nije uništena - sadržaj je nepouzdan - program je prekinut tokom obavljanja niza logički povezanih izmjena
  - vraćanje na ispravno stanje - pomoću dnevnika izmjena poništavaju se sve nepouzdane izmjene (sve izmjene koje su napravile nezavršene transakcije)

Dnevnik izmjena



Tipovi grešaka

1. Greške unutar transakcije (*transaction failure*) – greške koje su posljedica neplaniranog prekida transakcije
2. Kvar računarskog sistema (*system failure*) - baza nije fizički uništena
3. Kvar medija za pohranu (*media failure*) - fizički uništena baza

Slučaj 1 – pomoću dnevnika izmjena poništavaju se efekti transakcije, kao da transakcija nikada nije započela sa radom

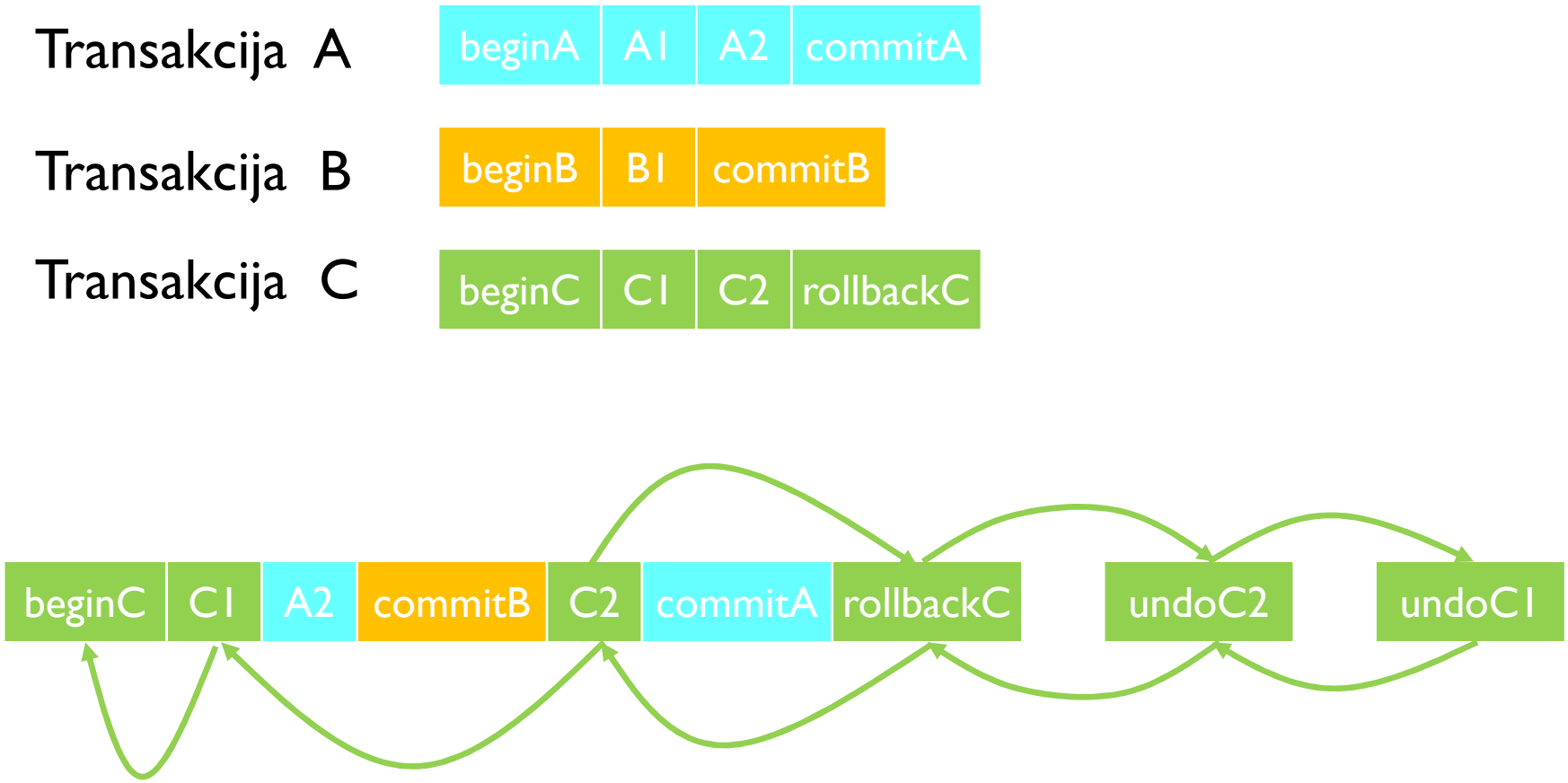
Slučaj 2 – transakcije koje su se obavljale u trenutku prekida se nakon ponovog pokretanja poništavaju

Slučaj 3 – baza podataka se obnavlja pomoću arhivske kopije i pripadajućeg dnevnika izmjena

Greške unutar transakcija

- U slučaju greške transakcije SUBP će poništiti efekte transakcije
- Dovesti bazu u stanje kao da transakcija nikada nije počela sa radom
- poništavanje izmjena - pretragom dnevnika unazad - nova vrijednost zapisa zamjenjuje se sa starom vrijednošću - sve dok se ne dođe do početka transakcije - BEGIN WORK (START TRANSACTION)

Dnevnik izmjena



Greške koje otkriva aplikacija

- Slučajevi u kojima aplikacija predviđa obavljanje naredbe ROLLBACK WORK

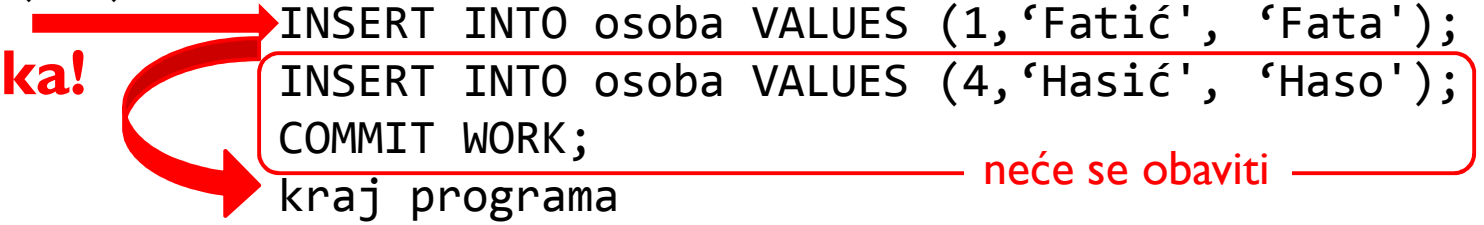
```
...
IF pom_saldo < 0 THEN
      ROLLBACK WORK;
ELSE
      COMMIT WORK;
END IF
...
```

Greške koje ne otkriva aplikacija

- Po završetku sesije SUBP automatski poništava sve nepotvrđene transakcije
  - Npr. ako se javi greška za koju aplikacija nije pretpostavila akciju - program (a u konačnici i sesija) završi na neplanirani način
  - u tim slučajevima SUBP automatski obavlja ROLLBACK WORK !
  - Poništavanje efekata transakcije
- Primjer: pokušaj unosa zapisa čiji ključ već postoji u bazi:

```
CREATE TABLE osoba (
   mbr INTEGER,
   prezime CHAR(20),
   ime CHAR(20)
   PRIMARY KEY (mbr)
);
```

```
početak programa
START TRANSACTION;
INSERT INTO osoba VALUES (1,'Mujić', 'Mujo');
INSERT INTO osoba VALUES (2,'Husić', 'Huso');
INSERT INTO osoba VALUES (1,'Fatić', 'Fata');
INSERT INTO osoba VALUES (4,'Hasić', 'Haso');
COMMIT WORK;
kraj programa
```



Način zapisivanja

- Koriste se spremnici (*buffer*):
  - Spremnik dnevnika
  - Spremnik baze podataka
- Sadržaj spremnika zapisuje se u dnevnik/bazu podataka:
  - Kada je spremnik popunjen ili
  - Kada SUBP izda nalog

Postizanje izdržljivosti (*durability*)

- Očuvanje izmjena u slučaju razrušenja neposredno nakon završetka transakcije
- Ako kvar nastane:
  - Nakon potvrđivanja i
  - Prije nego što su izmjene iz memorijskih spremnika prebačene u bazu podataka
- izmjene bi u trenutku kvara bile izgubljene, ALI:
  - procedura za ponovo pokretanje provest će promjene u bazi podataka
  - vrijednosti koje je potrebno zapisati u bazu podataka pronalaze se u odgovarajućim zapisima u dnevniku izmjena
- Sadržaj spremnika dnevnika mora biti zapisan u dnevnik na disku prije nego što završi procedura potvrđivanja transakcije – *write-ahead log rule*

Greške sistema

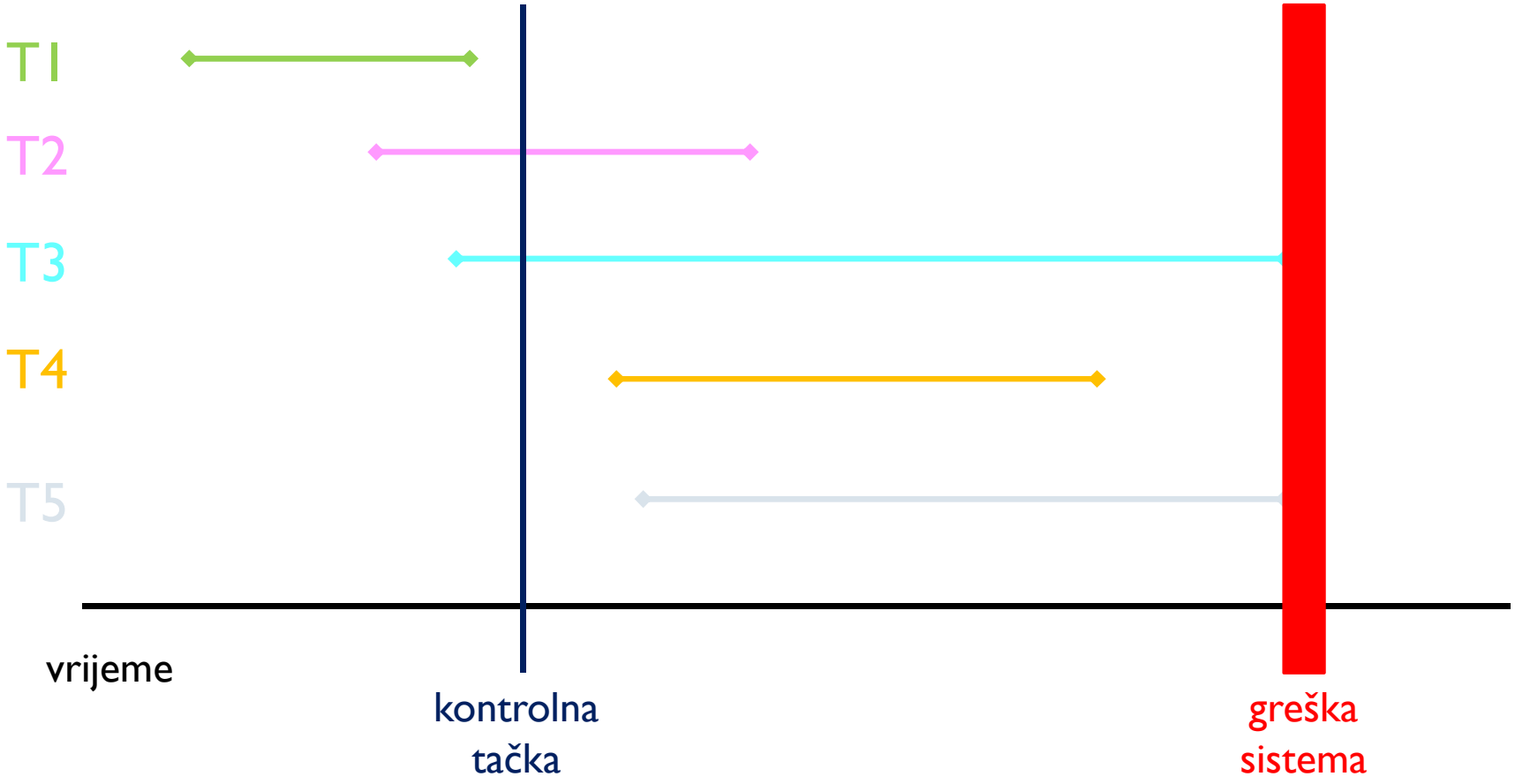
- Baza nije razrušena - sve transakcije koje su se odvijale u vrijeme kvara moraju biti poništene jer nisu kompletne!
- Pretraživanjem dnevnika od početka identifikuju se transakcije za koje postoji BEGIN (START) i ne postoji COMMIT ili ROLLBACK
  - takav postupak bi predugo trajao
  - uvode se kontrolne tačke (*checkpoint*)
  - u određenim intervalima (obično svakih 5 minuta) određuje se kontrolna tačka

Kontrolna tačka

1. pohrana sadržaja spremnika dnevnika (*log buffer*) u datoteku dnevnika
2. zapisivanje zapisa kontrolne tačke u datoteku dnevnika
3. zapisivanje adrese zapisa kontrolne tačke iz datoteke dnevnika u datoteku za ponovno pokretanje (*restart file*)
4. pohrana sadržaja spremnika baze podataka (*database buffer*) u bazu podataka

- Zapis kontrolne tačke sadrži:
  - listu svih aktivnih transakcija
  - za svaku transakciju - adresu najnovijeg zapisa u datoteci dnevnika

Primjer



Transakcije T3 i T5 treba poništiti

Transakcije T2 i T4 treba ponovo obaviti

Proces obnove

- Iz datoteke za ponovno pokretanje pročita se adresa posljednje kontrolne tačke
- Iz datoteke dnevnika pročita se zapis kontrolne tačke – lista transakcija koje su bile kativne u kontrolnoj tački i adrese njihovih zadnjih zapisa
- Stvara se:
  - lista za poništavanje - na početku sadrži sve transakcije koje su bile aktivne u kontrolnoj tački
  - lista za ponovo obavljanje - na početku je prazna
- Pretražuje se dnevnik od kontrolne tačke
  - transakcija za koju se pronađe BEGIN (START) dodaje se u listu za poništavanje
  - transakcija za koju se pronađe COMMIT prebacuje se iz liste za poništavanje u listu za ponovo obavljanje
- Ponovo se obavljaju transakcije iz liste za ponovo obavljanje
- Poništavaju se transakcije iz liste za poništavanje
- SUBP ne može prihvatiti niti jedan zahtjev dok se ne završi proces obnove!

Proces obnove

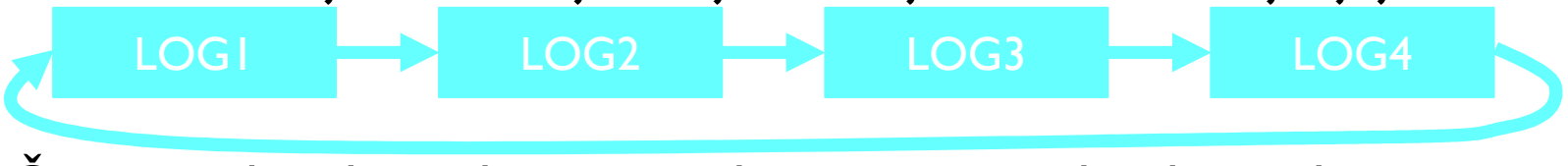
- Logika ponovnog obavljanja
  - Efekat ponovnog obavljanja, bez obzira koliko se puta obavljalo mora biti isti kao da je operacija obavljena tačno jednom!
  - redo (redo (redo (…..redo(x)))) = redo (x)
  - Ponovno obavljanje = Obnova unaprijed (*forward recovery*)
- Logika poništavanja
  - Efekat poništavanja, bez obzira koliko se puta obavljalo mora biti isti kao da je operacija obavljena tačno jednom!
  - undo (undo (undo (…..undo(x)))) = undo (x)
  - Poništavanje = Obnova unazad (*backward recovery*)

Kvarovi medija za pohranu

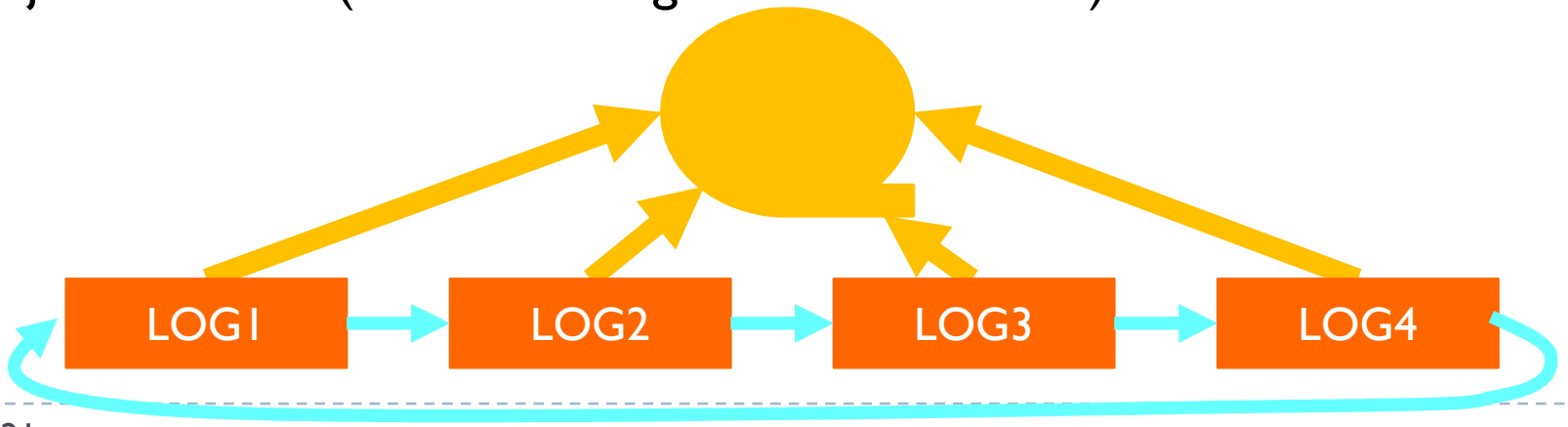
- baza je razrušena - npr. zbog kvara diska
  - obnova sadržaja baze pomoću najnovije arhivske kopije
  - pomoću najnovijeg dnevnika obavljaju se transakcije koje su bile provedene od trenutka arhiviranja
  - ako je najnovija arhivska kopija "pokvarena"
    - uzima se predzadnja arhivska kopija
    - dnevnik izmjena od predzadnje arhive do zadnje arhive
    - dnevnik izmjena nastalih nakon zadnje arhive
- Preporuke:
  - čuvati najmanje tri posljednje arhive i pripadne dnevnike
  - dnevnik se ne nalazi na istom disku na kojem je baza podataka

Ciklička izmjena logičkih dnevnika

- Dnevnik može biti vrlo velik – započinje u trenutku pokretanja arhiviranja i aktivan je do sljedećeg arhiviranja
- Dnevnici su ključni za obnovu – kako ih očuvati?
  - Čim prije treba njihov sadržaj pohraniti na "sigurno mjesto"
- Dnevnik se dijeli na manje odsječke koji se ciklički izmjenjuju



- Čim se jedan dnevnik popuni - kopira se na arhivski medij
- Dnevnik se mora nalaziti na disku sve dok su transakcije sadržane u njemu aktivne (da bi se omogućio ROLLBACK)



Ciklička izmjena logičkih dnevnika

- U slučaju da su svi dnevnici puni - obustavljaju se sve akcije i sistem čeka da se napravi sigurnosna kopija
  - vremenski preduge transakcije, prevelike transakcije
    - kad su dnevnici popunjeni npr. 70% - "prvi znak za uzbunu" (*Long Transaction High Water Mark*) – sistem ne prihvaća nove transakcije
    - kad su dnevnici popunjeni npr. 85 ili 90% - "drugi znak za uzbunu" (*Long Transaction Exclusive Access High Water Mark*) - sve aktivne transakcije se poništavaju
- Kod pogreške medija koristi se arhivska kopija + dnevnici s arhivskih medija

Obnova uz zadovoljenje zahtjeva za visokim stepenom dostupnosti

- "ogledanje" podataka (*mirroring*)
- arhiviranje tokom obavljanja transakcija (*online backup*)
- inkrementirano arhiviranje
- fizički dnevnik (*physical log*)

“Ogledanje”

- Uz primarno područje postoji i slika podataka “u ogledalu” (*mirror*)
- Promjene se provode istovremeno u primarnom i “ogledalnom” području
- U slučaju pogreške u jednom od područja
  - nastavak rada na ispravnom području
  - nalog za ponovo kreiranje (“popravljanje”) “ogledalnog” područja
  - nakon “popravka” područja se sinhroniziraju
- “ogledanje” se može izvoditi pod kontrolom *hardware*-a
  - RAID - *Redundant Arrays of Inexpensive Disks*
- “ ogledanje” pod kontrolom SUBP-a – na nivou baze podataka određuje se koji dio baze podataka će se „ogledati”

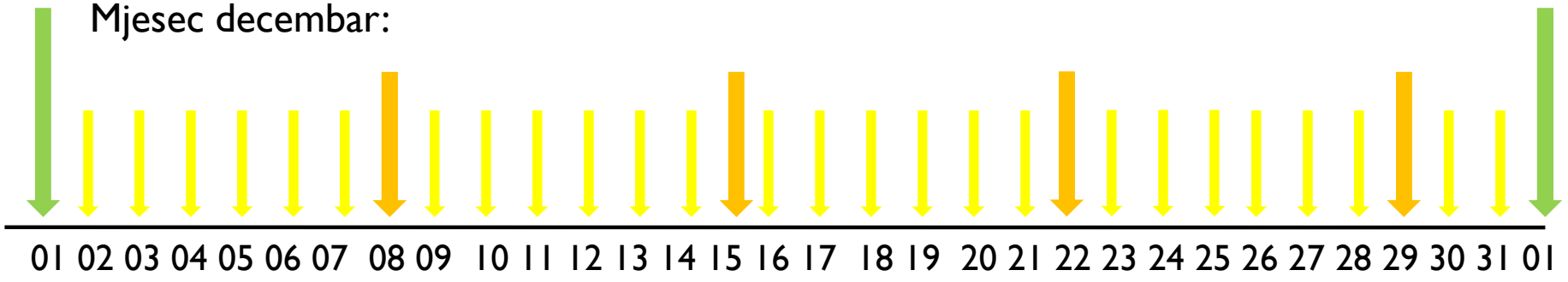
“Ogledanje” na nivou baze podataka

- Omogućava finiju granulaciju ogledalnog područja
  - Mogu se ogledati samo dijelovi baze podataka – relacije koje uvijek moraju biti dostupne
- Visoka dostupnost – u slučaju kvara jednog područja dostupni su podaci u drugom
- Skalabilnost – podjela opterećenja
- “Ogledanje” ne može pomoći pri poništavanju transakcija niti kod kvara čitavog sistema
- “Ogledanje” je samo dodatni mehanizam za postizanje visoke dostupnosti
- Ne može zamijeniti arhiviranje i vođenje dnevnika

Arhiviranje

- On-line arhiviranje (*On-line Backup*)
  - stvaranje arhivske kopije tokom rada korisnika – izvođenja transakcija
  - stanje baze podataka pohranjeno u arhivskoj kopiji odgovara stanju kakvo je bilo u trenutku pokretanja arhiviranja
- Inkrementirano arhiviranje
  - arhiviranje baze podataka dodatno opterećuje sistem i usporava rad korisnika
  - želi se skratiti trajanje i obim arhiviranja
  - inkrementirano arhiviranje omogućava stvaranje arhiva različitih nivoa:
    - nivo 0 – kopija čitave baze podataka – npr. jednom mjesečno
    - nivo 1 – sedmična arhiva – sadrži promjene nastale nakon arhive nivoa 0
    - nivo 2 – dnevna arhiva - sadrži promjene nastale nakon arhive nivoa 1

Inkrementirano arhiviranje



arhiva nivoa 0

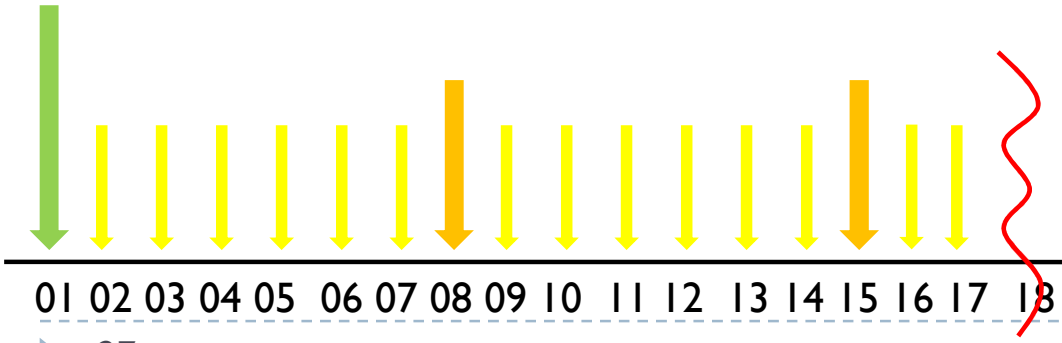
arhiva nivoa 1

arhiva nivoa 2

Obnova:

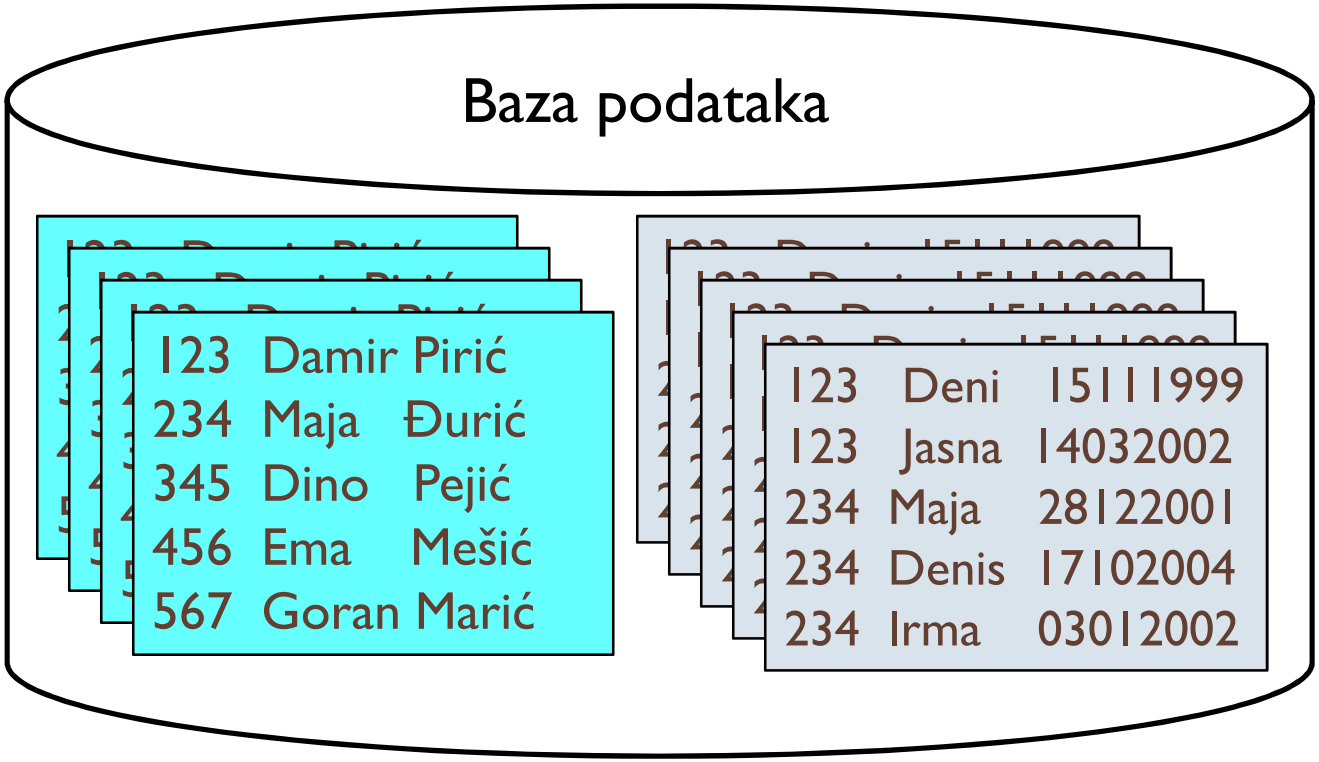
1. Arhiva nivoa 0 od 1. decembra
2. Arhiva nivoa 1 od 15. decembra
3. Arhiva nivoa 2 od 17. decembra
4. Logički dnevnik koji je započeo nakon arhive nivoa 2 od 17. decembra

Ako se 18. decembra desi kvar na disku



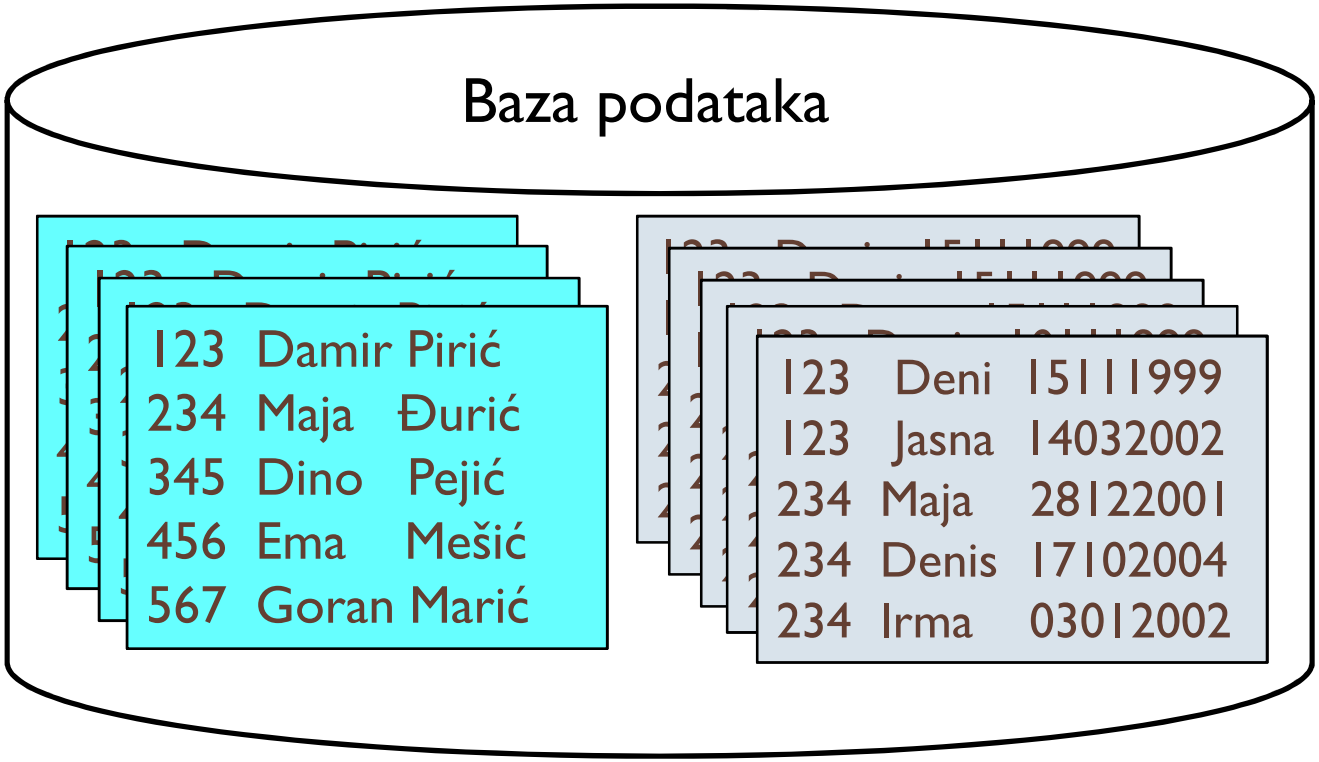
Fizički dnevnik

- stvara se nakon kontrolne tačke
- pohranjuju se kopije stranica s diska (iz baze podataka) prije obavljanja izmjena (*Before Image*)
- omogućava rekonstrukciju stanja u bazi podataka kakvo je bilo u posljednjoj kontrolnoj tački prije pogreške
- ostvaruje se fizička konzistentnost podataka
- Kod ponovog pokretanja sistema:
  - provjerava se postoji li neki sadržaj u fizičkom dnevniku – ako postoji - bilo je izmjena nakon posljednje kontrolne tačke - pokreće se brza obnova:
    - sve stranice fizičkog dnevnika pohranjuju se na svoje lokacije
    - obavljaju se izmjene iz logičkog dnevnika počevši od kontrolne tačke, poništavaju se transakcije koje nisu završile prije pogreške



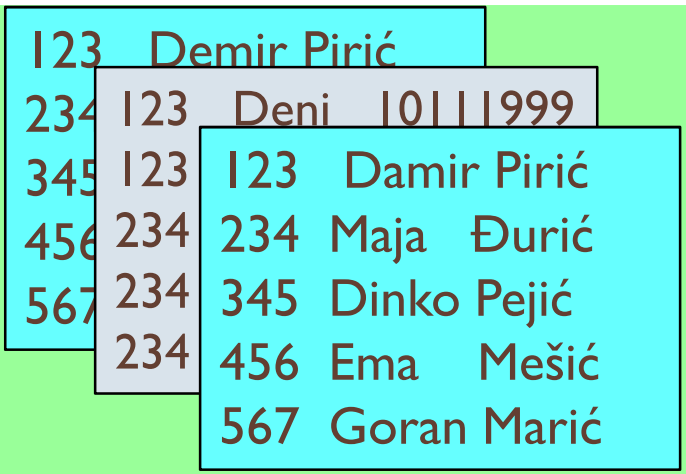
Tabele:
osoba(mbr, ime, prez)
dijete(mbr, imedj, datrod)

Nakon neke kontrolne tačke fizički dnevnik je prazan

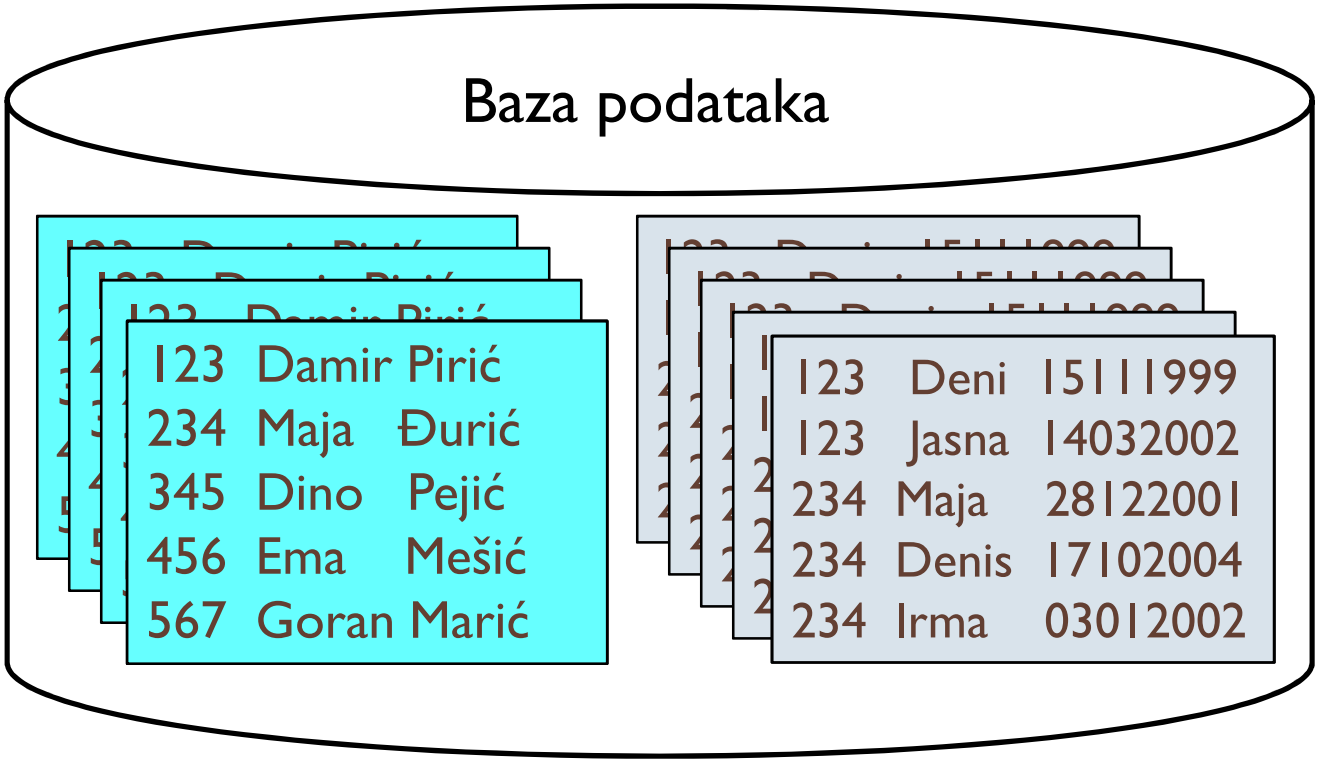


Tabele:
osoba(<u>mbr</u>, ime, prez)
dijete(<u>mbr</u>, <u>imedj</u>, datrod)

Prije svake izmjene stranica na kojoj će doći do promjene kopira se u datoteku fizičkog dnevnika (struktura – stack – LIFO)

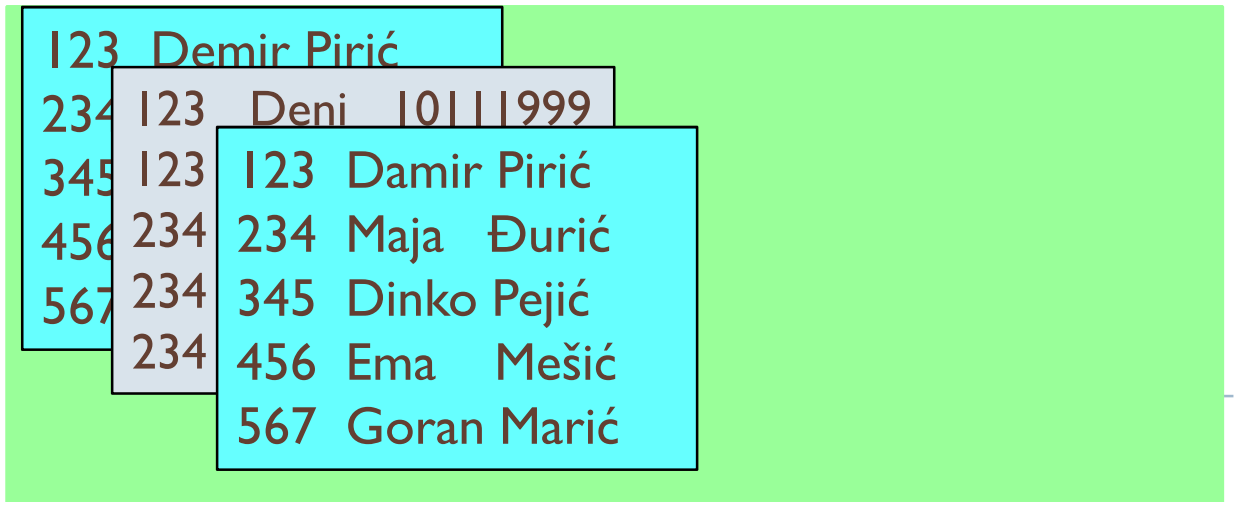


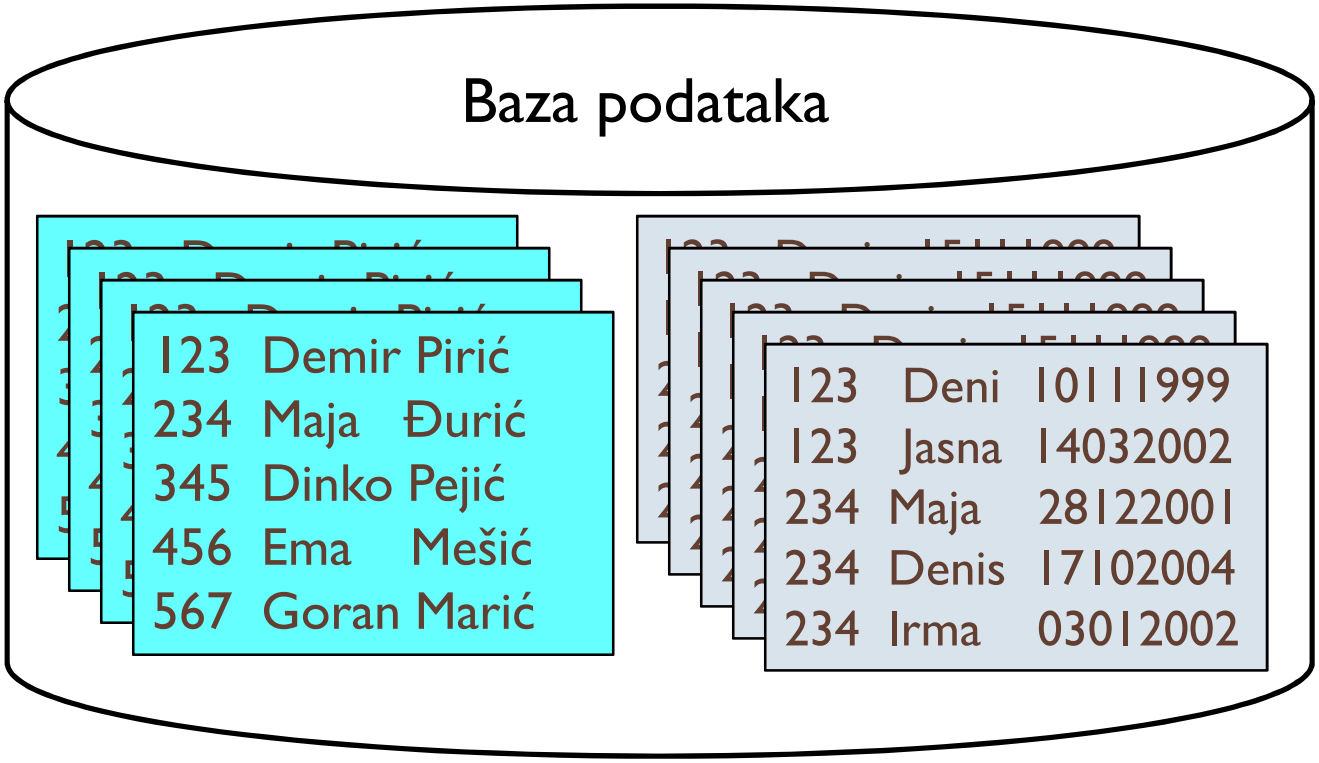

```
UPDATE osoba set ime = ‘Damir'
  WHERE mbr = 123;
UPDATE dijete SET datrod =
'15.11.1999'
  WHERE mbr = 123
  AND imedj = ‘Deni';
UPDATE osoba set ime =‘Dino'
  WHERE mbr = 345;
```



Tabele:
osoba(<u>mbr</u>, ime,
prez)
dijete(<u>mbr</u>, <u>imedj</u>,
datrod)

Obnova – stranice iz fizičkog dnevnika kopiraju se u bazu podataka





Tabele:
osoba(<u>mbr</u>, ime,
prez)
dijete(<u>mbr</u>, <u>imedj</u>,
datrod)

Nakon završetka kopiranja sadržaja iz datoteke fizičkog dnevnika u bazu podataka – sadržaj baze podataka je vraćen u stanje kakvo je bilo u kontrolnoj tački