Oporavak od otkaza

BP 2 2021/22

OTKAZI

Tipovi otkaza:

- Otkazi trasakcija – transakija/e se ne mogu izvršiti
  - Logička greška - prekoračenje neke od dozvoljenih vrednosti, narušavanje integriteta,
  - Interna greška – mrtvi čvor
- Sistemski otkaz u sistemu
  - Softverski problem - problem sa OS-om ili DMBS implementacijom (deljenje nulom)
  - Hardverski problem - prestanak električnog napajanja
- Otkaz memorijskog medija - u disku na kome je baza podataka, npr. oštećenje glava diska (nema pomoći od strane DBMS-a osim preuzimanja arhivirane baze).

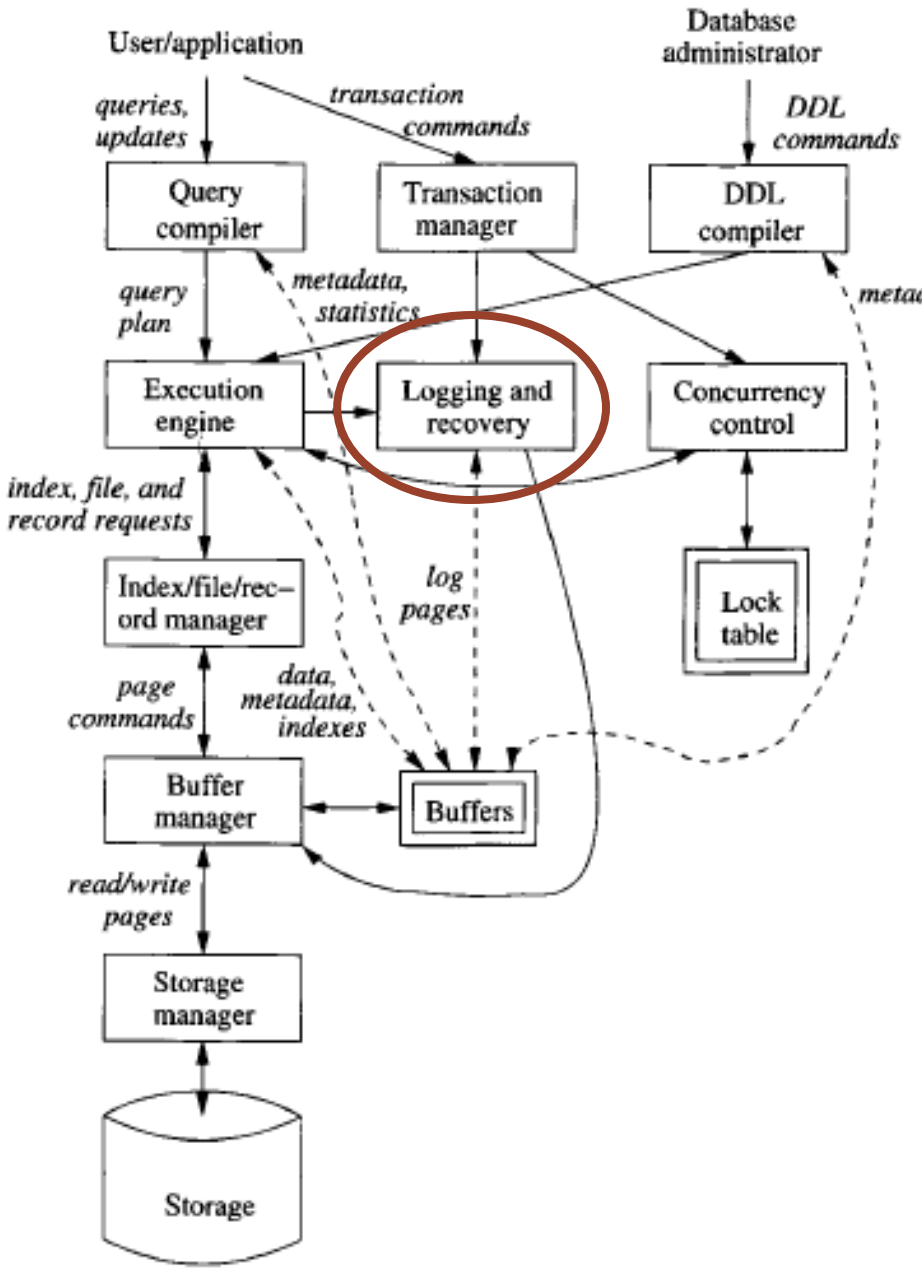
OPORAVAK OD OTKAZA

U slučaju otkaza - greške u transakciji, u sistemu ili u mediju (disku) DBMS mora da oporavi sadržaj baze podataka da bi obezbedio:

- Zadovoljenost (privremeno narušenih) uslova integriteta baze
- Atomičnost trasakcija
- Trajnost transakcija

DBMS mora da obezbedi trajnost efekata potvrđenih transakicja poništi delimične promene napravljene od strane poništenih transakcija.

UNDO – proništavanje efekata poništenih ili nezavršenih transakcija.
REDO – ponovno izvršavanje potvrđenih transakcija.



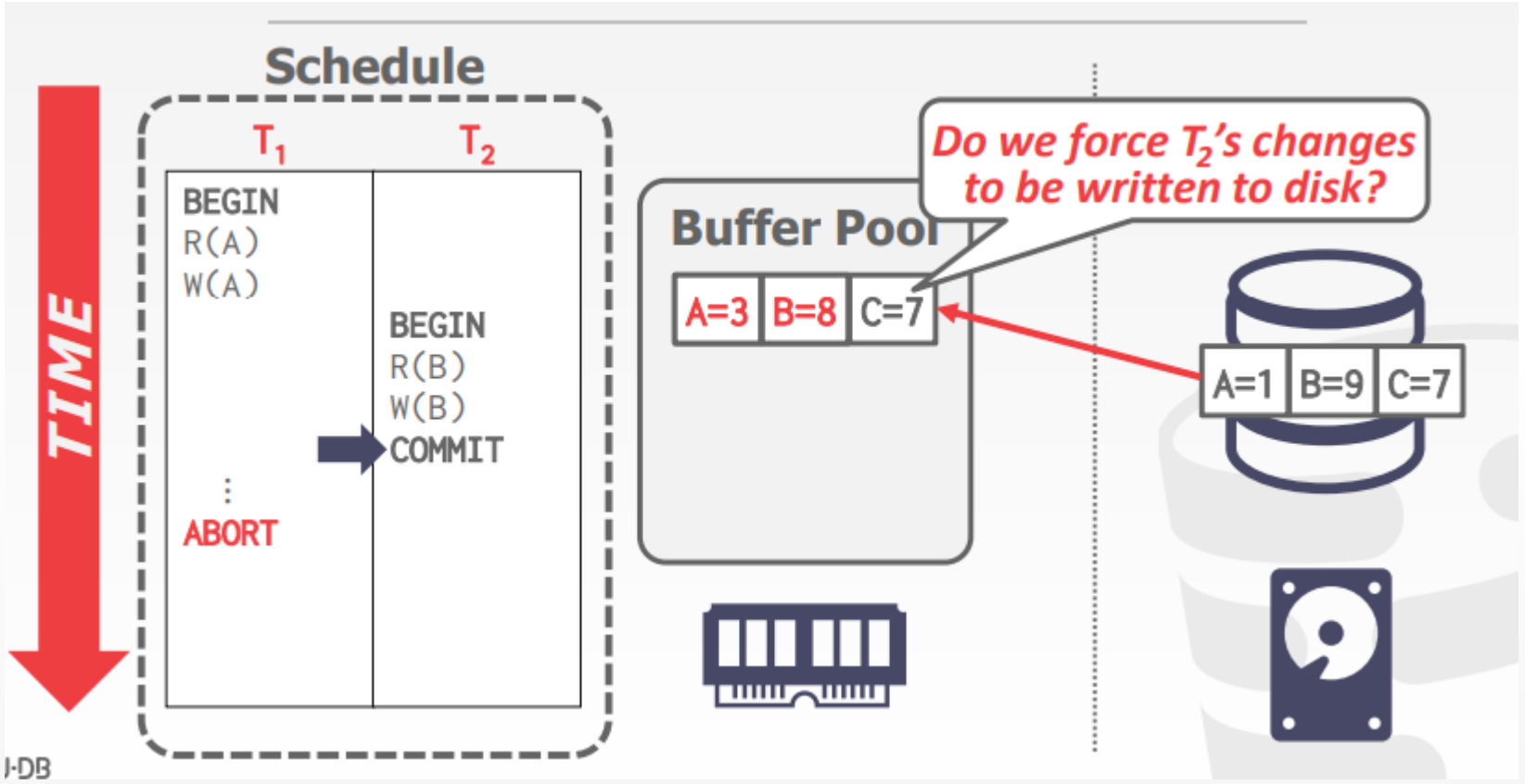
Algoritmi za oporavak

Algoritmi za oporavak podrazumevaju dve vrste radnji:

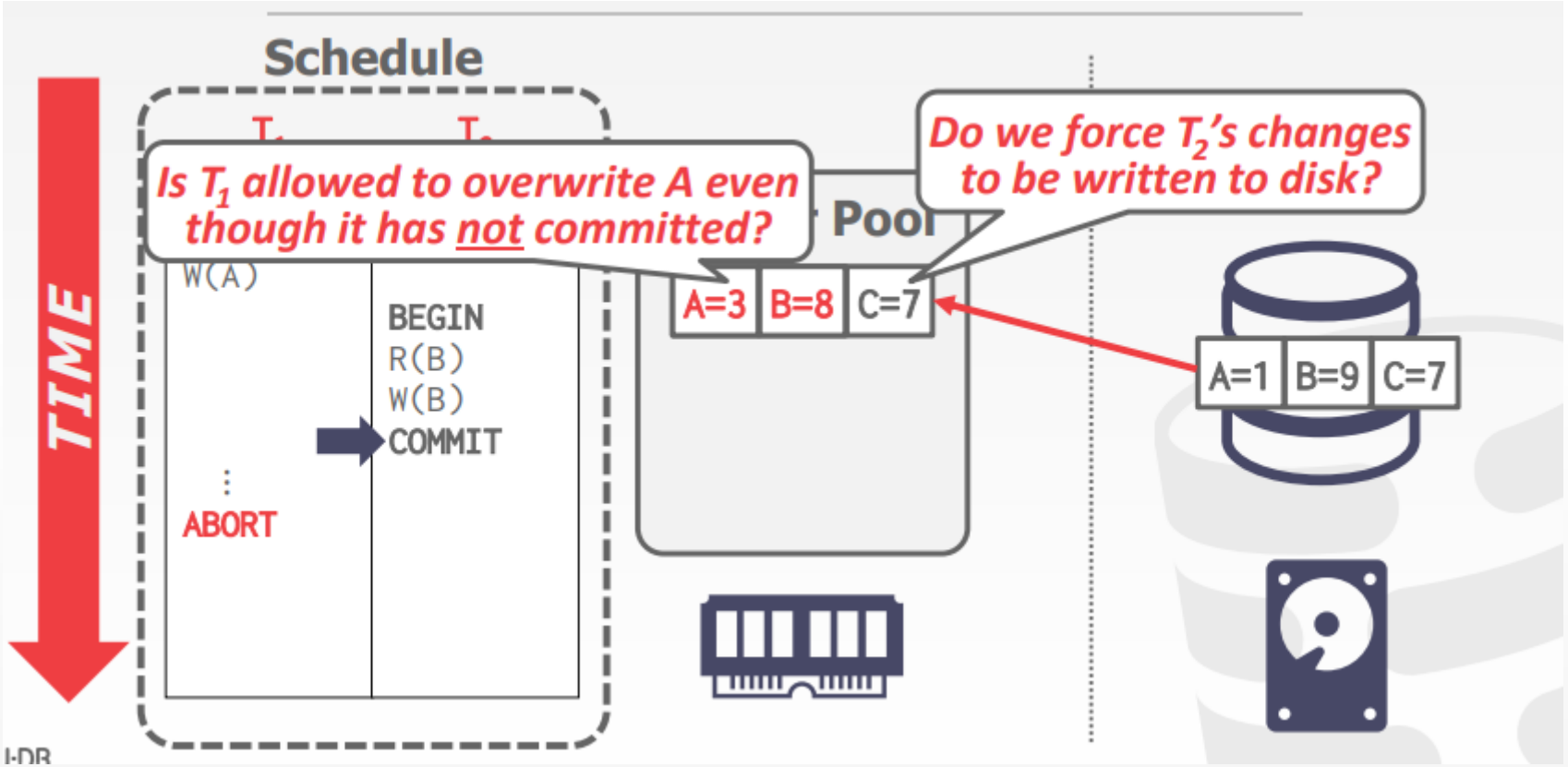
- Radnje koje se vrše tokom regularnog izvršavanja transakcija koje će osigurati da se DBMS može oporaviti od otkaza.
- Radnje nakon otkaza kojima se baza podataka vraća u konzistentno stanje.

Protokoli tokom izvršavanja transakcija

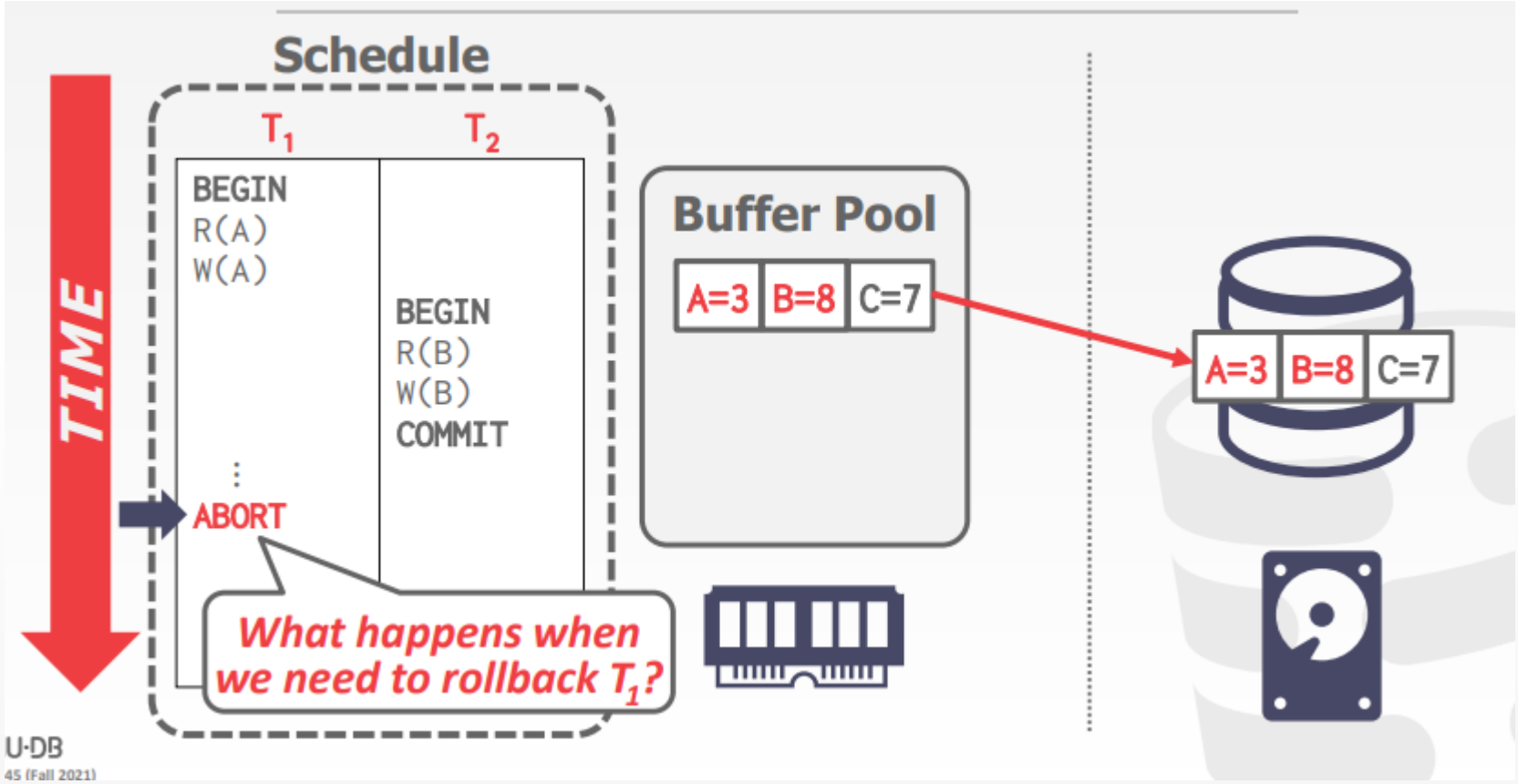
BAFER PUL POLITIKE



BAFER PUL POLITIKE



BAFER PUL POLITIKE



BAFER PUL POLITIKE

Politika prepisivanja (steal policy)

Da li je dozvoljeno nepotvrđenim transakcijama da prepišu potvrđenu vrednost na stalnom skladištu?

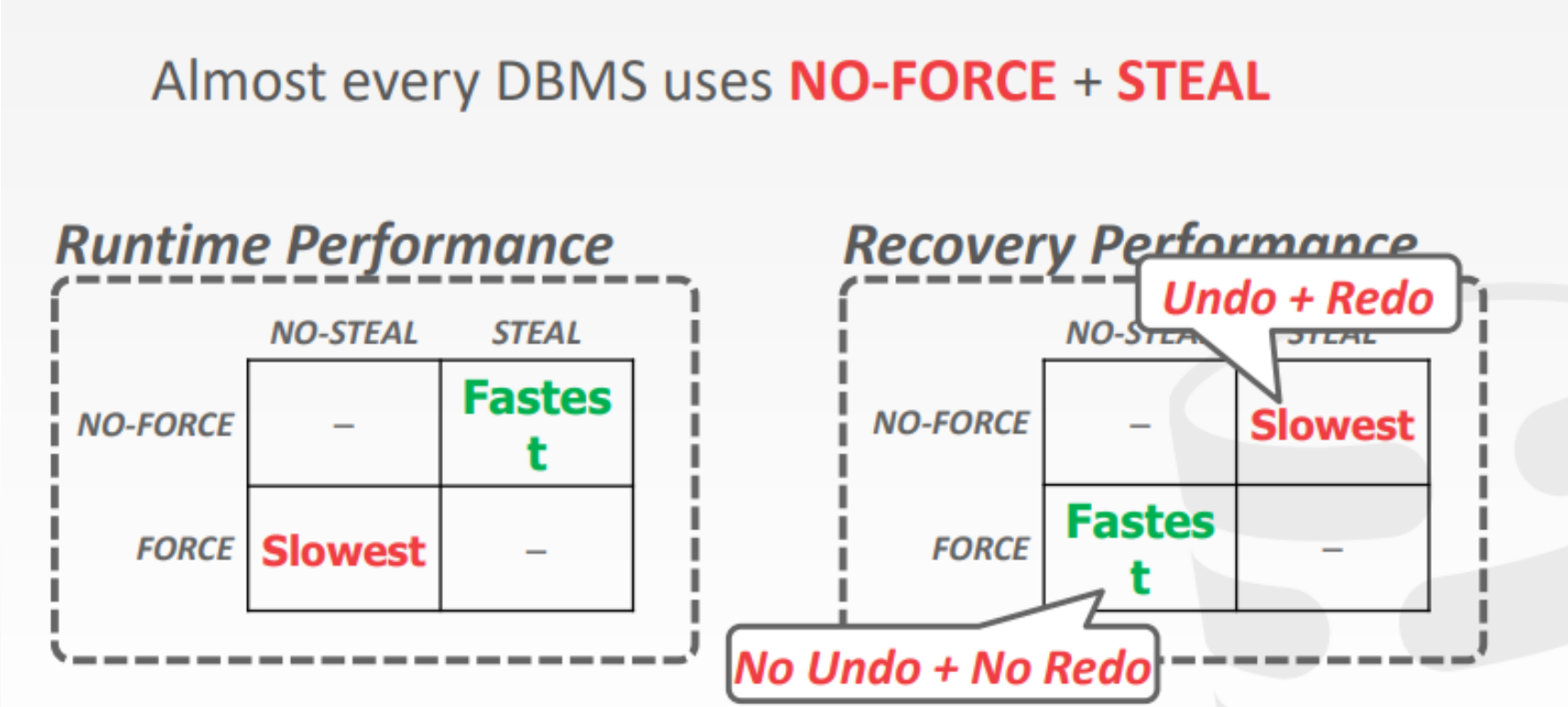
STEAL: Dozvoljeno.
NO-STEAL: Nedozvoljeno.

Politika forsiranog upisa (force policy)

Da li DBMS zahteva da sve promene koje je transakcija napravila budu upisane u trajno skladište pre nego što se transakcija proglasi potvrđenom?

FORCE: Obavezno.
NO-FORCE: Nije obavezno.

BAFER PUL POLITIKE



WAL – Write-Ahead Log protokol

Shadowing – izmenjene vrednosti se drže na kopiji koja se čiva na disku.
Kada se trasakcija potvrdi, izmenjene strane se preuzimaju kao stalne

WAL

- LOG - Datoteka evidencije
  - Održava odvojeno od datoteke sa podacima.
  - Sadrži opise promena koje transakicja vrši.
  - Čuva se na stalnom skladištu.
  - Sadrži dovoljno informacija za izvršavanje UNDO/REDO operacija.

WAL protokol

- Upis COMMIT sloga u log datoteku i upis svih ažuriranja te transakcije u bazu – dve odvojene radnje
- WAL protokol obezbeđuje da se izvrše obe
  - prvo se odgovarajući slog fizički upisuje u log datoteku,
  - pa se zatim podaci upisuju iz bafera podataka u bazu.
- Ako dođe do pada sistema posle upisa COMMIT sloga u log datoteku a pre nego što je sadržaj bafera podataka prepisan u bazu, restauriranje iz sistemskog loga REDO logikom.

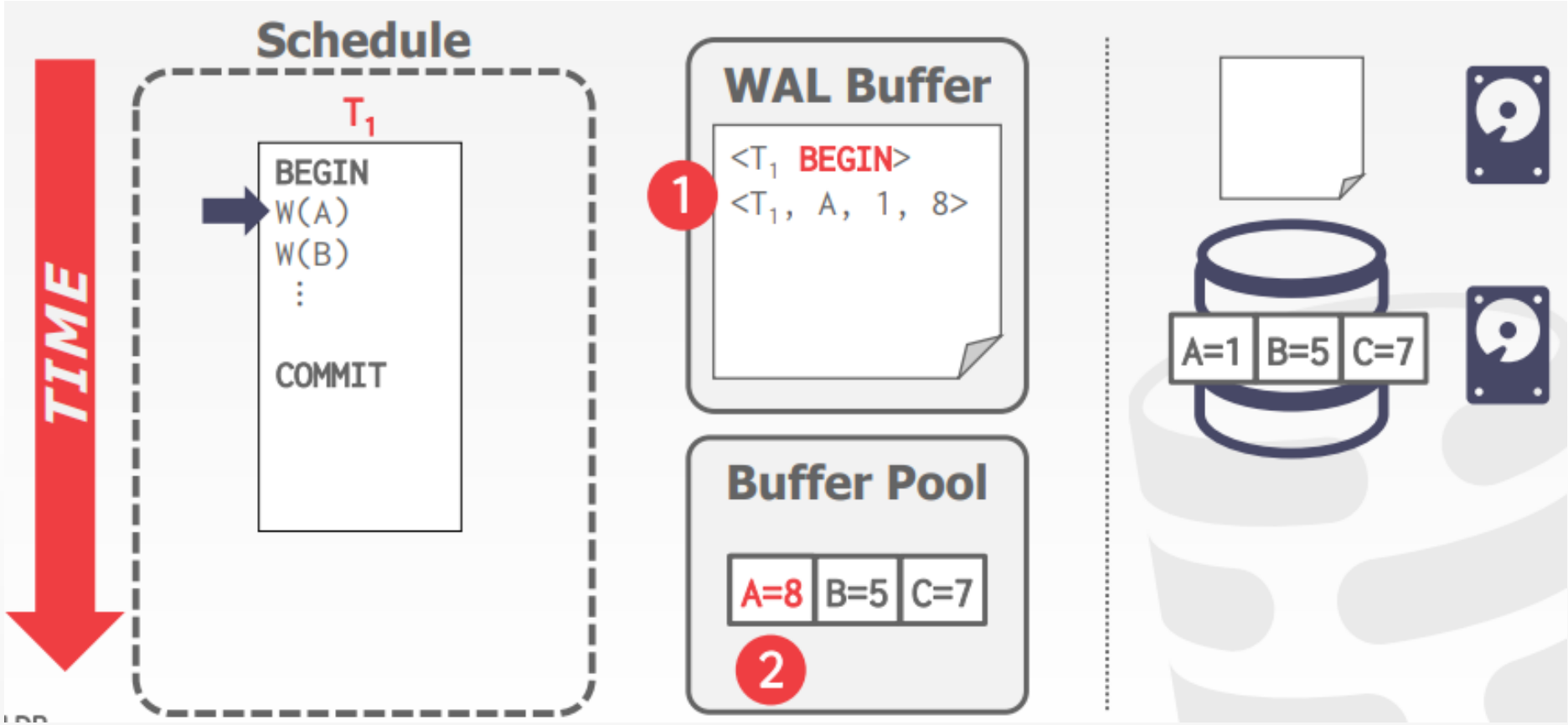
WAL protokol

- DBMS beleži sve zapise dnevnika ažuriranja (LOG) u radnoj memoriji (pozadinski proces bafer menadžera)
- Svi zapisi dnevnika koji se odnose na ažuriranu stranicu se upisuju u stalno skladište pre same stranice sa izmenjenim podacima.
- Transakcija se ne smatra podvrđenom sve dok se svi zapisi dnevnika ne upišu u trajno skladište.

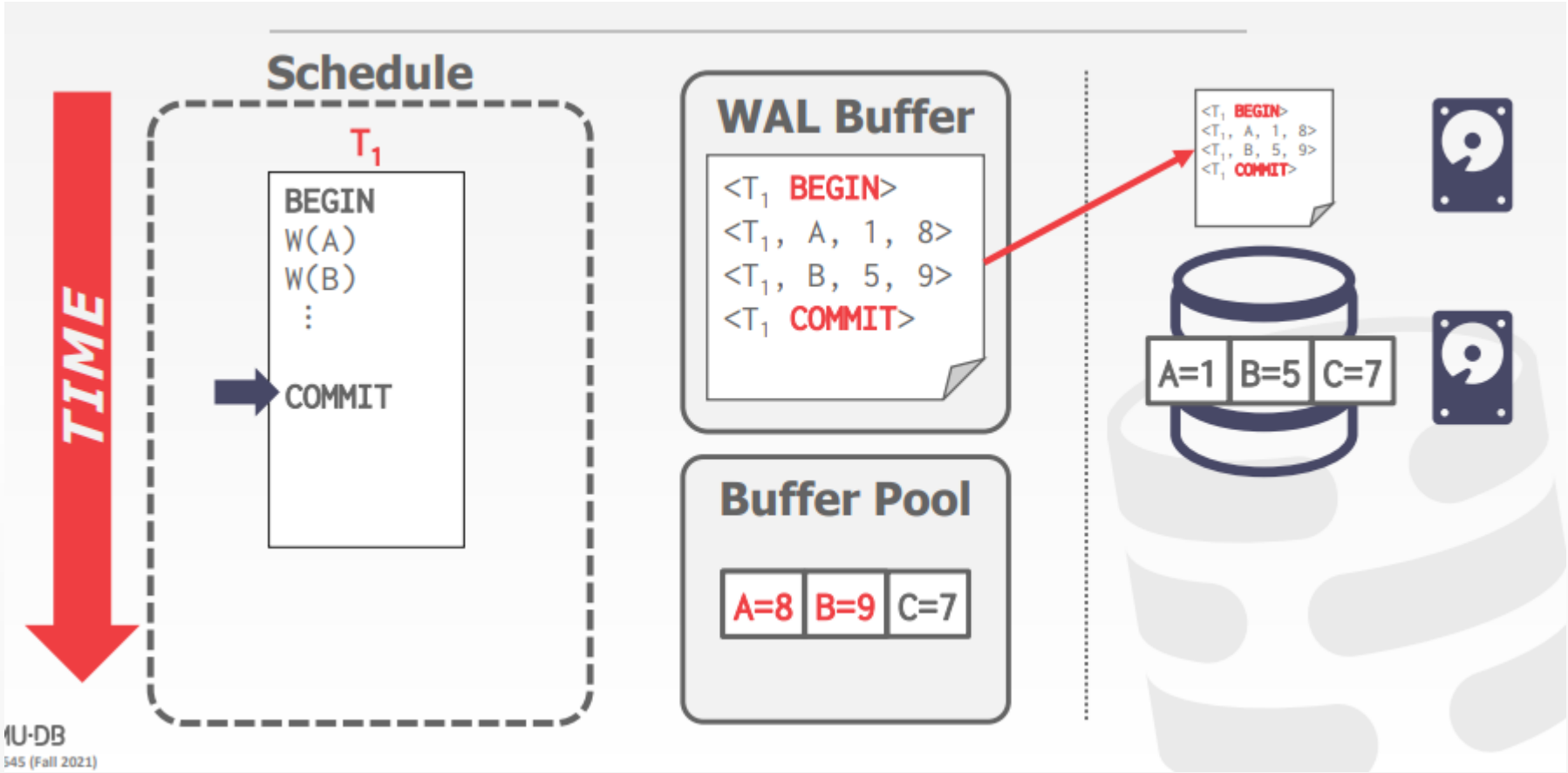
WAL zapisi

- Svaki zapis sadrži informaciju o izmeni jednog objekta:
  - ID transakcije
  - ID objekta
  - Stara vrednost
  - Nova vrednost
- Pri ponovnom startovanju sistema, posle pada sistema, moguće je prema sadržaju log datoteke identifikovati
  - neuspele transakcije – kandidate za poništavanje, i
  - uspele transakcije – kandidate za ponovno izvršavanje.

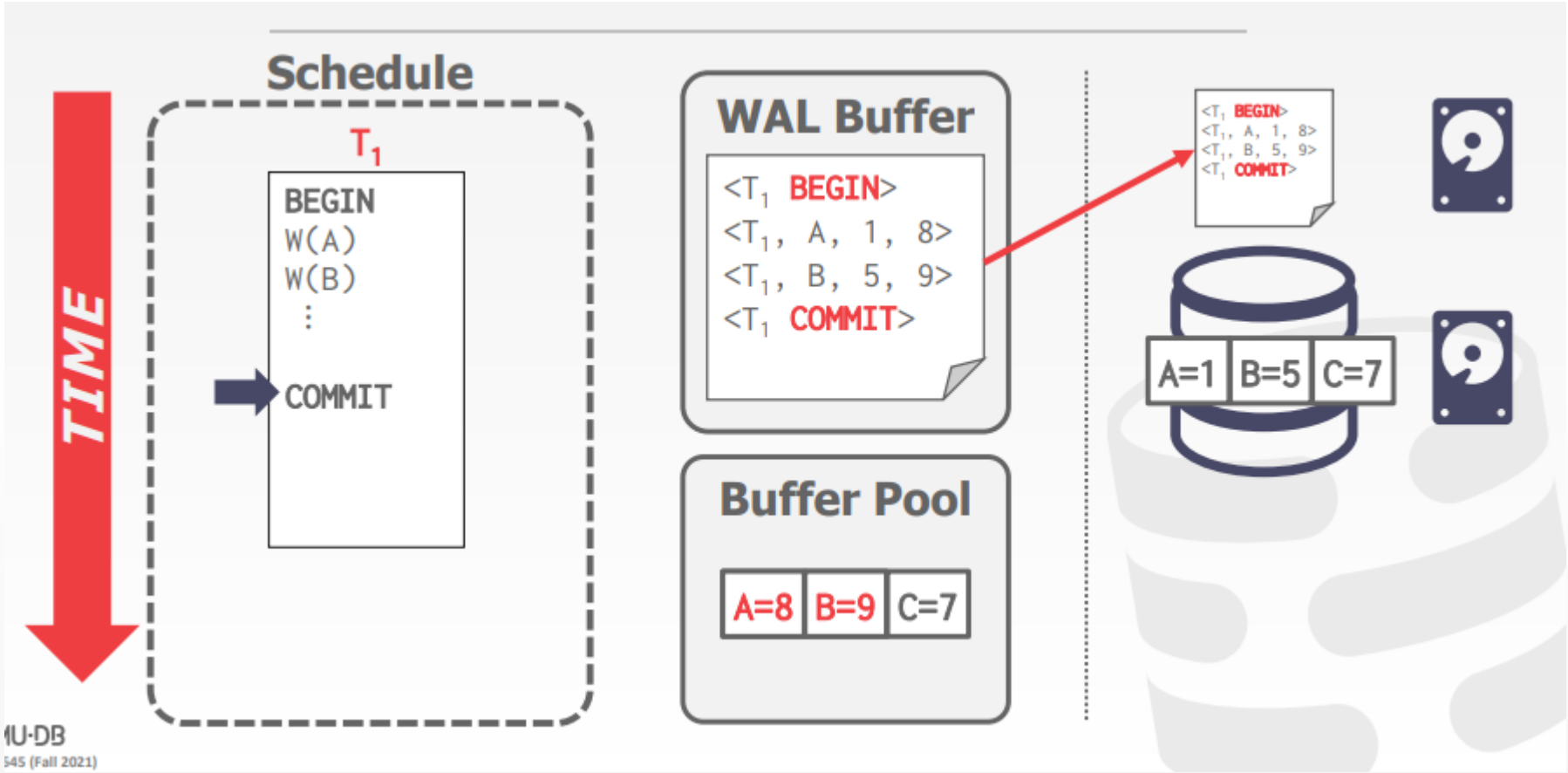
WAL primer


Schedule
T1
BEGIN
W(A)
W(B)
COMMIT
TIME
WAL Buffer
<T1 BEGIN>
<T1, A, 1, 8>
1
Buffer Pool
A=8
B=5
C=7
2
A=1
B=5
C=7

WAL primer

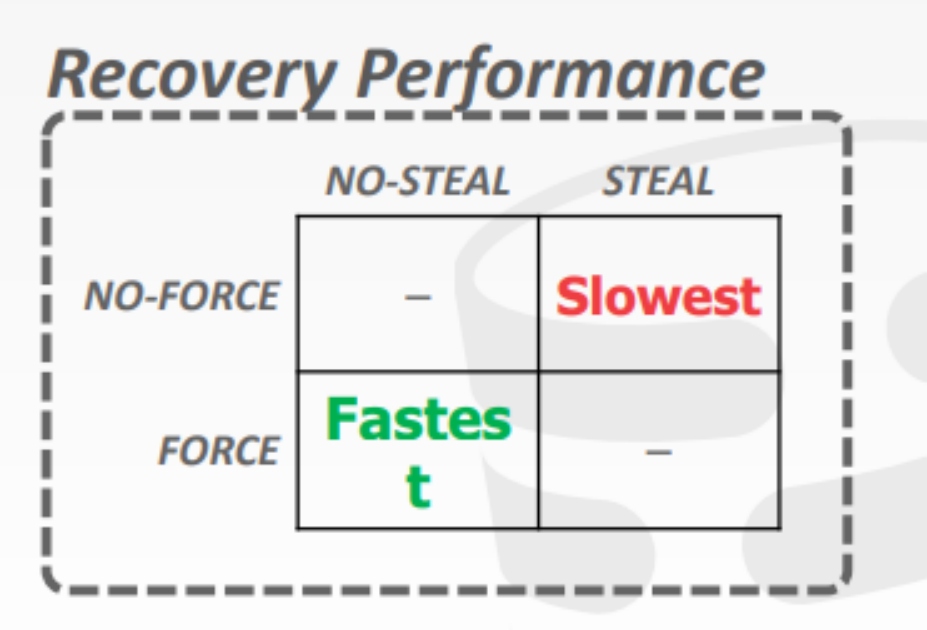

Schedule
T1
BEGIN
W(A)
W(B)
COMMIT
TIME
WAL Buffer
<T1 BEGIN>
<T1, A, 1, 8>
<T1, B, 5, 9>
<T1 COMMIT>
Buffer Pool
A=8
B=9
C=7
<T1 BEGIN>
<T1, A, 1, 8>
<T1, B, 5, 9>
<T1 COMMIT>
A=1
B=5
C=7
U-DB
545 (Fall 2021)

WAL primer


Schedule
T1
BEGIN
W(A)
W(B)
COMMIT
TIME
WAL Buffer
<T1 BEGIN>
<T1, A, 1, 8>
<T1, B, 5, 9>
<T1 COMMIT>
Buffer Pool
A=8
B=9
C=7
<T1 BEGIN>
<T1, A, 1, 8>
<T1, B, 5, 9>
<T1 COMMIT>
A=1
B=5
C=7

WAL implementacija

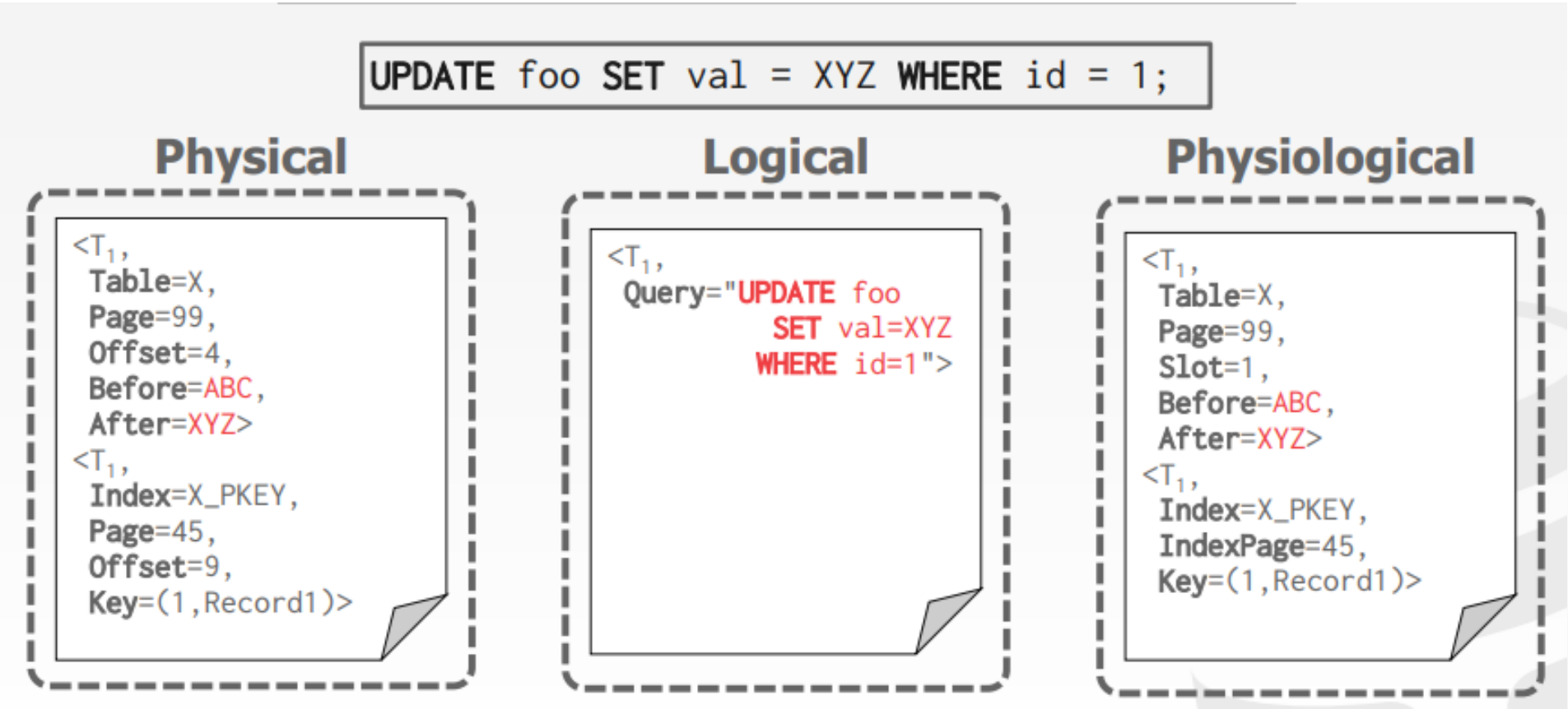
- Kada se zapisi dnevnika beleže na disk?
  - Nakon potvrđivanja transakcije.
  - Višestruki commit – kada se napune stranice dnevnika
- Kada se beleže prljave strane sa podacima?
  - Nakon svakog ažuriranja
  - Nakon potvrđivanja



Zapisi u dnevniku

- Fizički zapisi
  - Beleže se lokacije u bazi gde su izvršene izmene
- Logiški zapisi
  - Beleže se operacije
- Hibridni zapis (najčešće)
  - Beleži se stranica i broj slota nad kojim je izvršena promena

Zapisi u dnevniku



Checkpoint – tačka pamćenja

- Beleženje izmena se vrši povremeno - tačka pamćenja - bez obzira da li su baferi puni.
- Fizički upis ažuriranih podataka uspešno kompletirane transakcije garantuje se tek u tački pamćenja.
- Fizički upis ažuriranih podataka uspešno kompletirane transakcije garantuje se tek u tački pamćenja.

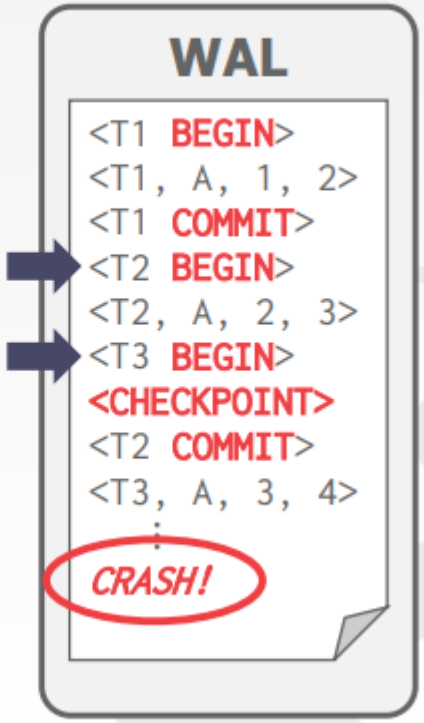
Checkpoint – tačka pamćenja

Any txn that committed before the checkpoint is ignored (T_1).

T_2 + T_3 did not commit before the last checkpoint.

→ Need to redo T_2 because it committed after checkpoint.

→ Need to undo T_3 because it did not commit before the crash.



Oporavak – osnovni algoritam

```
BEGIN
    naći slog poslednje tačke pamćenja u log datoteci
            (iz datoteke ponovnog startovanja);
    IF u poslednjoj tački pamćenja nema aktivnih transakcija
      i slog tačke pamćenja je poslednji slog u log datoteci,
      oporavak je završen
    ELSE
        BEGIN
            formirati dve prazne liste transakcija, "uspele" i "neuspele";
            sve transakcije aktivne u poslednjoj tački pamćenja
                staviti u listu neuspelih;
            čitati redom log datoteku od tačke pamćenja do kraja:
                kada se naiđe na slog "početak transakcije",
                    dodati transakciju listi neuspelih;
                kada se naiđe na slog "kompletirana transakcija",
                    dodati (premestiti) tu transakciju listi uspelih;
            čitati unazad log datoteku (od kraja)
                i poništiti akcije i efekte neuspelih transakcija;
            čitati log datoteku od poslednje tačke pamćenja unapred
                i ponovo izvršiti uspele transakcije
        END
END.
```