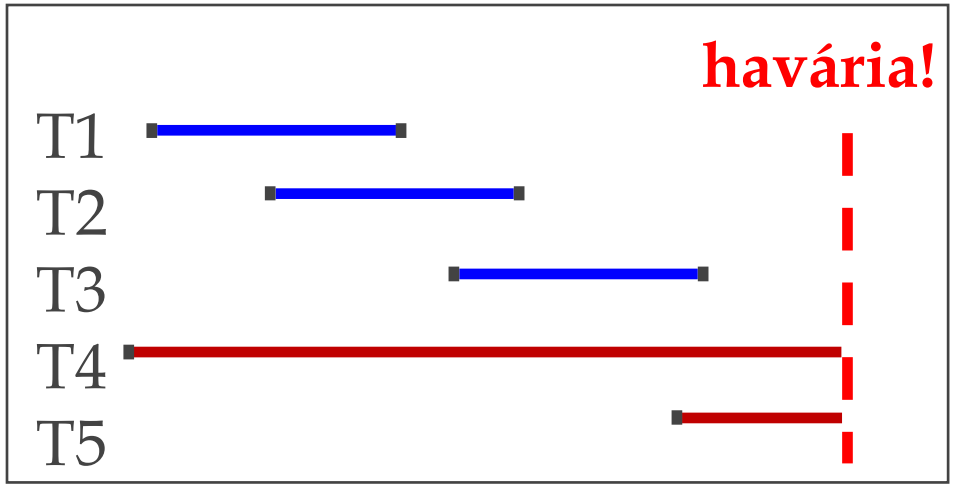
# Zotavenie z chýb (Crash Recovery)

Kapitola 20

# ACID vlastnosti

- Atomic - Atomickosť (buď sa vykoná celá alebo sa databáza musí dostať do stavu, akoby sa nevykonala vôbec)
- Consistency - Konzistentnosť (po transakcii musí byť DB znova konzistentná – zachované integritné obmedzenia)
- Isolation - Nezávislosť (sú nezávislé od iných transakcií)
- Durability - Trvanlivosť (po potvrdení vykonania sú zmeny uložené)
- **Manažér zotavenia** garantuje atomickosť a trvanlivosť transakcii.
  - Atomickosť: Transkacia môže byť zrušená (“Rollback”).
  - Trvanlivosť: Čo sa udeje, ak DBMS prestane bežať?

- Očakávané chovanie po reštarte:
  - T1, T2, T3 majú byť trvanlivé.
  - T4, T5 majú byť zrušené (akoby sa nikdy nespustili).



# Politiky manažéra buffra

- NO STEAL: zmenené stránky ostávajú v buffri až do ukončenia transakcie (EOT)
  - Čo ak nemám dosť RAMky?
- STEAL: zmenené stránky môžu byť na disk zapísané kedykoľvek
  - Ako zabezpečím atomickosť?
  - Musíme si pamätať predchádzajúcu verziu stránky, aby sme vedeli vrátiť prechádzajúci stav
- FORCE: Všetky transakciou zmenené stránky sú na konci transakcie (EOT) zapísané na disk
  - Časovo náročné...
- NO FORCE: Nie je nutné ukladať všetky zmenené stránky na disk pred skončením transakcie
  - Ako zachováme trvanlivosť?
  - Zapisujeme nutné minimum, aby sme po páde vedeli operáciu zopakovať

| | No Steal | Steal |
|---|---|---|
| Force | jednoduché | |
| No Force | | žiadané |

# Write Ahead Log (WAL)

- Každá zmena databázového objektu je najprv zapísaná do logu ešte pred tým než sa zmena zapíše na disk
- Ak buffer manažér uloží zmenenú stránku na disk (STEAL) pred tým, ako je príslušný log zapísaný na disk, nemusíme vedieť urobiť UNDO
- Ak zmenená stránka ostane v pamäti (NO FORCE) a systém padne pred tým ako sa táto zmena zapíše do logu, nevieme urobiť REDO
- Pravidlá WAL:
  - Log musí byť zapísaný na disk **pred tým** ako sa príslušná dátová stránka zapíše na disk → garantuje atomickosť
  - Log musí byť zapísaný **pred tým** ako sa transakcie commitnú → garantuje trvanlivosť

# Manažér zotavenia (recovery manager)

- Musí sa postarať o to, aby sa systém po zlyhaní zotavil.
- Po zlyhaní systému manažér vykonáva nasledujúce kroky:
  - **Analýza** - identifikuje zmeny, ktoré neboli uložené na disk (dirty pages) a transakcie, ktoré prebiehali keď systém zlyhal.
  - **Redo** – zopakuje všetky akcie od istého bodu a obnoví DB do stavu v akom bol v tom bode.
  - **Undo** - vráti všetky akcie transakcie, ktoré sa nekomitli, aby v DB boli iba komitnuté transakcie
- Jeden z najpoužívanejších algoritmov na zotavenie je ARIES a má 3 princípy:
  - Write-ahead logovanie
  - Zopakovanie histórie počas Redo
  - Logovanie zmien počas Undo

# Logovanie

- Logujeme informácie pre REDO a UNDO pre každú zmenu dát
  - Zapisujeme sekvenčne (na koniec logu) ideálne na separátny disk.
  - Zapisujeme iba rozdiely, takže zmeny viacerých stránok môžu vojsť na jedinú logovaciu stránku
- Logovací záznam, ktorý popisuje zmeny obsahuje
  - <ID_transakcie, ID_stránky, offset, dĺžka, staré dáta, nové dáta>
  - plus ďalšie riadiace informácie
    - Typ logovacieho záznamu = UPDATE
    - ID logovacieho záznamu LSN
    - ID predchádzajúceho logovacieho záznamu v rámci transakcie prevLSN
- Ďalšie typy logovacích záznamov:
  - Commit
  - Abort
  - End (koniec abortu alebo commitu)
  - CLR: zrealizované UNDO

# Log

- Súbor so zoznamom akcií vykonaných databázou.
- Najaktuálnejšia časť zvaná log tail sa nachádza v hlavnej pamäti.
- Každý záznam dostane id, ktorý sa volá log sequence number (**LSN**)
- Každá stránka v DB si pamätá LSN najnovšieho záznamu v logu, ktorá zodpovedá času poslednej zmeny na tejto stránke (**pageLSN**)
- Databáza si pamätá posledné LSN zapísané na disk (**flushedLSN**)
  - Stránka môže byť zapísaná na disk iba ak pageLSN ≤ flushedLSN
- Typy záznamov v LOGu:
  - Update stránky
  - Commit
  - Abort
  - End (koniec abortu alebo commitu)
  - CLR: zrealizované UNDO

# Pomocné pamäťové štruktúry

- Tabuľka transakcií:
  - Jeden záznam pre každú transakciu.
  - Záznam obsahuje id transakcie, lastLSN (LSN najnovšieho log záznamu pre danú transakciu) a stav (beží/comitnutá/zrušená)
- Tabuľka zmenených stránok:
  - Záznam o každej zmenenej stránke v buffer poole
  - Záznam obsahuje recLSN (číslo LSN prvého logovacieho záznamu, ktorá spôsobila zmenu stránky)

# Vytváranie checkpointov

- Checkpointy sa vytvárajú periodicky, aby sa minimalizoval čas zotavenia systému po páde.
- Slúži na vytvorenie "snapshotu" z aktuálneho stavu DBMS
- ARIES algoritmus používa fuzzy checkpoint → v logu sa vytvorí:
  - begin_checkpoint záznam: zodpovedá času, v ktorom sa vytvorí snapshot
  - end_checkpoint záznam: snapshot záznam s tabuľkou transakcií a tabuľkou zmenených stránok
  - Na disku sa ešte uloží LSN posledného checkpoint záznamu (*master* záznam).

- Commit
- Abort
- End (koniec abortu alebo commitu)
- CLR: zrealizované UNDO

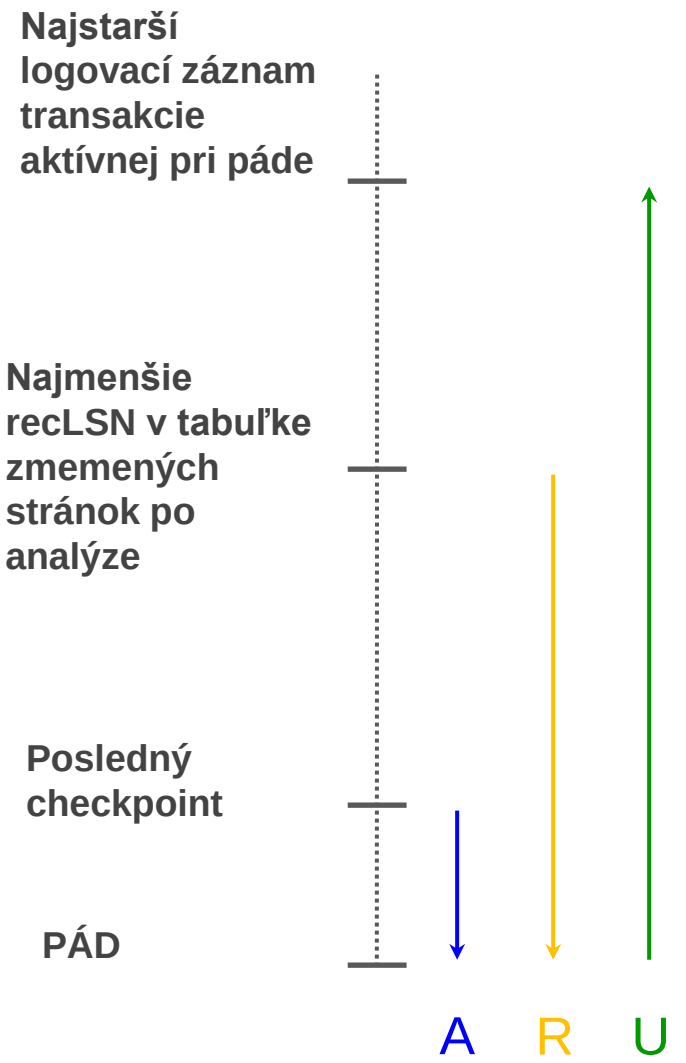
# Transakcia s commitom

- Zapíšeme commit záznam do logu
- Všetky logy až po lastLSN transakcie sa zapíšu na disk
  - Zabezpečíme že flushedLSN ≥ lastLSN.
  - Zápis logu je sekvenčný a zapíše mnoho logovacích záznamov na jednu stránku
- metóda commit() je ukončená
  - Zmenené dátové stránky sa zapísať nemusia
- Zapíšeme end záznam do logu

# Transakcia s rollbackom

- Rollback vyžiadaný transakciou
  - Bez pádu systému
- Chceme čítať log v opačnom poradí a robiť UNDO operácie
  - Aby sme mohli urobiť UNDO, musíme dať zámok na dáta
  - Vezmeme lastLSN transakcie z tabuľky transakcií.
  - Nasledujeme predchádzajúce logovacie záznamy cez hodnotu prevLSN.
  - Pred realizáciou UNDO operácie zapíšeme nahradzovací (CLR) logovací záznam
    - Pre korektné zotavenie, ak systém padne počas UNDO operácie
      - Pokračujeme v logovaní aj počas UNDO
    - CLR má extra atribút: undonextLSN - ukazuje na ďalšie LSN v rámci UNDO (t.j. prevLSN záznamu na ktorom práve robíme UNDO).
    - CLR záznamy sa nikdy nespracúvajú v rámci UNDO (ale môžu byť znovu vykonané (REDO) keď opakujeme históriu po páde: garantuje atomicitu!)
  - Na konci UNDO zapíšme záznam end pre danú transakciu.

# Zotavenie sa po zlyhaní systému




- Začneme od checkpointu (nájdenom cez master záznam)
- Tri fázy:
  - Analýza toho, ktoré transakcie boli commitnuté po checkpointe
  - REDO – zopakujeme všetky akcie
  - UNDO – zrušíme zmeny necommitnutých transakcií

# Fáza analýzy

- Hlavná úloha: zrekonštruovať tabuľku zmenených stránok a tabuľku transakcií, v čase pádu
- podúloha: nájsť bod v logu odkiaľ bude prebiehať Redo fáza
  - Nájde stránky v buffer pooli, ktoré sú dirty v čase pádu
  - Identifikuje transakcie, ktoré prebiehali počas pádu a ktorého zmeny musia byť vrátené
- Prechádza log dopredu od checkpointu
    - End záznam: Vymaž transakciu z tabuľky transakcií
    - Ostatné záznamy: Pridaj transakciu do tabuľky transakcií, nastav lastLSN=LSN, zmeň stav transakcie podľa logovacieho záznamu
    - Update záznam: Ak stránka nie je v tabuľke zmenených stránok, pridaj ju a nastav jej recLSN=LSN.

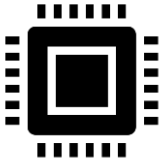
# Fáza Redo

- **Hlavná úloha**: zrekonštruovať buffer pool do podoby v čase pádu
- Opakuje históriu:
  - Znovu zrealizuj *všetky* zmeny (aj tie, vykonané abortovanými transakciami), opakuj aj zmeny z CLR
- Číta log dopredu od záznamu s najmenším číslom recLSN v tabuľke zmenených stránok. Opakuje všetky update a CLR akcie okrem prípadov:
  - Stránka nie je v stránke zmenených stránok
  - Stránka je v stránke zmenených stránok ale má recLSN > LSN
  - pageLSN (na disku) ≥ LSN.
- Opakovanie akcie:
  - Znovu vykoná to, čo je v logu.
  - Nastaví pageLSN na LSN.
  - Žiadne ďalšie logovanie

# Fáza Undo

- **Hlavná úloha**: abort nedokončených transakcií
- Na rozdiel od ostatných postupuje spätne z konca.
- Začína s tabuľkou tranzakcií z prevej fázy.
- Pracuje s množinou ToUndo = { *L* | *L* je lastLSN nedokončenej transakcie}
- **Opakuj:**
  - Vyber najväčšie LSN z ToUndo.
  - Ak ide o CLR a undonextLSN == NULL
    - Zapíš End záznam tejto transakcie
  - Ak ide o CLR a undonextLSN != NULL
    - Pridaj undonextLSN do ToUndo
  - Inak ide o update.  Zapíš CLR, vráť stav pred updateom, pridaj prevLSN do ToUndo.
- **pokiaľ** ToUndo **nie je prázdne**

# Príklad zotavenia po páde

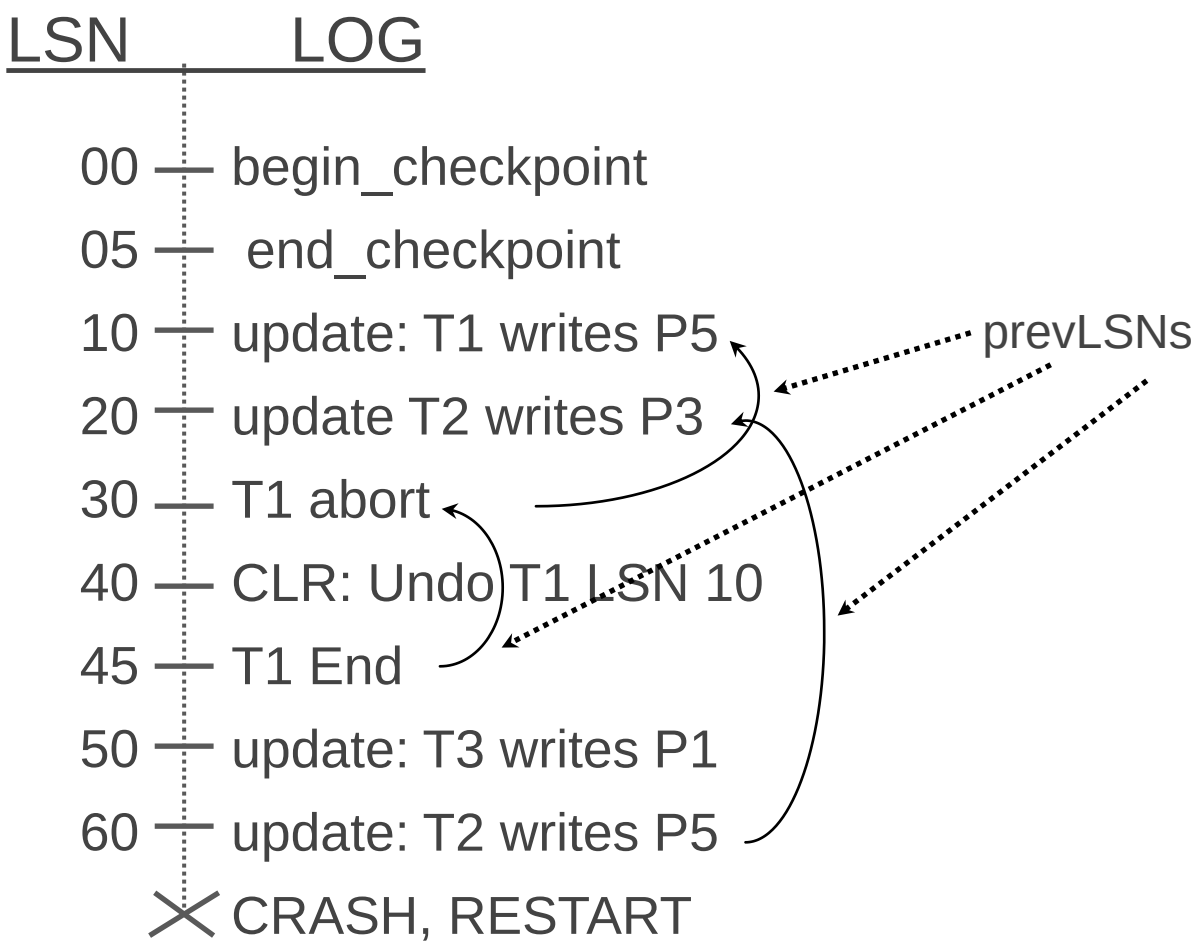


Tabuľka transakcií
- lastLSN
- status

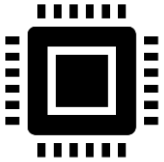
Tabuľka zmenených stránok
- recLSN

flushedLSN

ToUndo



# Príklad pádu počas reštartu

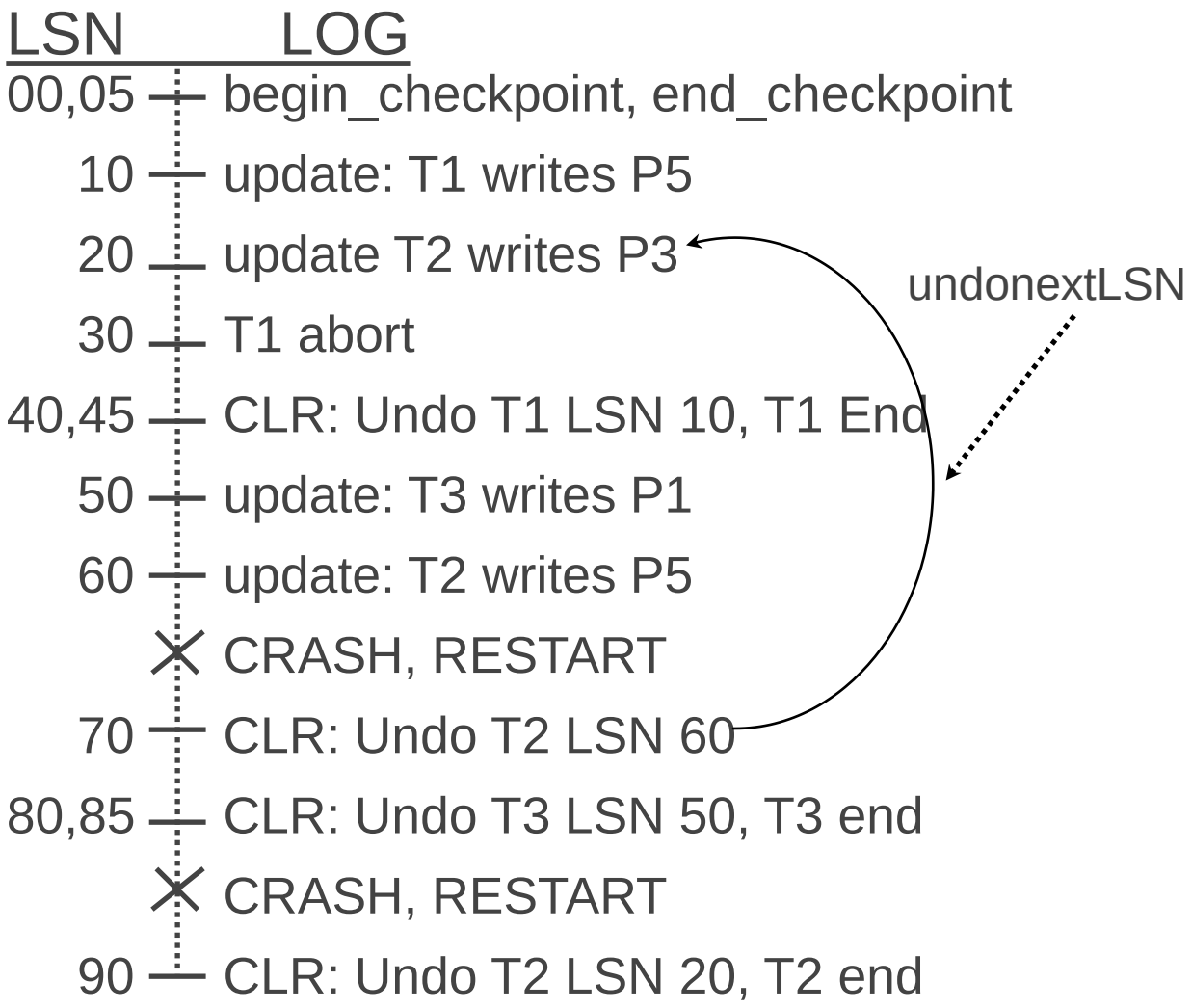


Tabuľka transakcií
- lastLSN
- status

Tabuľka zmenených stránok
- recLSN

flushedLSN

ToUndo



# Poznámky k zotavovaniu

- Čo sa stane ak systém padne počas analýzy alebo počas REDO?
- Ako obmedziť množstvo práce vykonávanej počas REDO fázy?
  - Priebežné ukladanie zmenených stránok na disk
- Ako obmedziť množstvo práce vykonávanej počas UNDO fázy?
  - Zabráňte vzniku dlhotrvajúcich transakcií