Gruppe 10 / 48

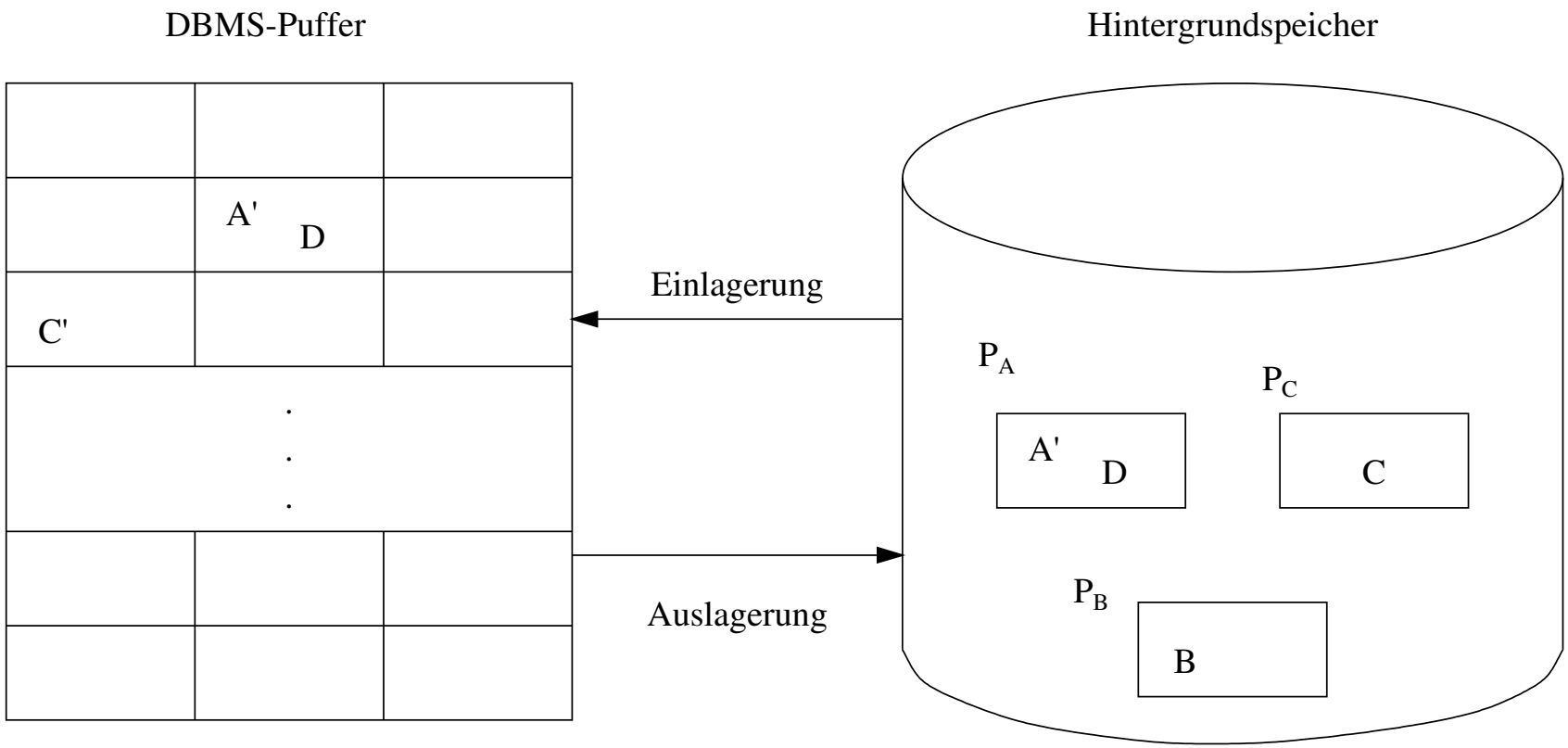
Tutorübung zu Grundlagen:Datenbanken (WS 18/19)

Michael Schwarz

Institut für Informatik
Technische Universität München

06.02 / 07.02.2019

Speicherhierarchie



Speicherhierarchie(2)

- Ersetzungsstrategien von Puffer-Seiten
  - $\neg$*steal*: Ersetzung von Seiten, die von einer noch aktiven Transaktion modifiziert wurden, ausgeschlossen
  - *steal*: Jede nicht fixierte Seite ist prinzipiell ein Kandidat für die Ersetzung, falls neue Seiten eingelagert werden müssen
- Einbringen von Änderungen abgeschlossener TAs
  - *force*-Strategie: Änderungen werden zum Transaktionsende auf den Hintergrundspeicher geschrieben
  - $\neg$*force*-Strategie: geänderte Seiten können im Puffer verbleiben und später zurückgeschrieben werden

Recovery

- Wichtige Aufgabe eines DBMS ist das Verhindern von Datenverlust durch Systemabstürze
- Die zwei wichtigsten Mechanismen des Recovery sind:
  - Sicherungspunkte (Backups)
  - Log-Dateien

Recovery(2)

- Ein *Sicherungspunkt* ist ein Schnappschuß des Datenbankinhalts zu einem bestimmten Zeitpunkt
- In einer *Log-Datei* werden alle Änderungen an der Datenbasis mitprotokolliert
- Offensichtlich sollten Sicherungspunkte und Log-Dateien nicht auf der gleichen Maschine gespeichert werden ...

ARIES-Protokoll

- ARIES-Protokoll ist weit verbreitetes Protokoll zur Fehlerbehandlung in DBMSen
- Log-Datei enthält:
  - Redo-Information: gibt an, wie Änderungen nachvollzogen werden können
  - Undo-Information: beschreibt, wie Änderungen rückgängig gemacht werden können

Struktur der Log-Einträge

[LSN,TA,PageID,Redo,Undo,PrevLSN]

- Redo:
  - Physische Protokollierung: After-Image
  - Logische Protokollierung: Code mit dem aus dem Before-Image das After-Image erzeugt werden kann
- Undo:
  - Physische Protokollierung: Before-Image
  - Logische Protokollierung: Code mit dem aus dem After-Image das Before-Image erzeugt werden kann

Struktur der Log-Einträge(2)

- *LSN* (*Log Sequence Number*),
  - eine eindeutige Kennung des Log-Eintrags
  - *LSNs* müssen monoton aufsteigend vergeben werden,
  - die chronologische Reihenfolge der Protokolleinträge kann dadurch ermittelt werden
- *TA*
  - Transaktionskennung der Transaktion, die die Änderung durchgeführt hat

Struktur der Log-Einträge(3)

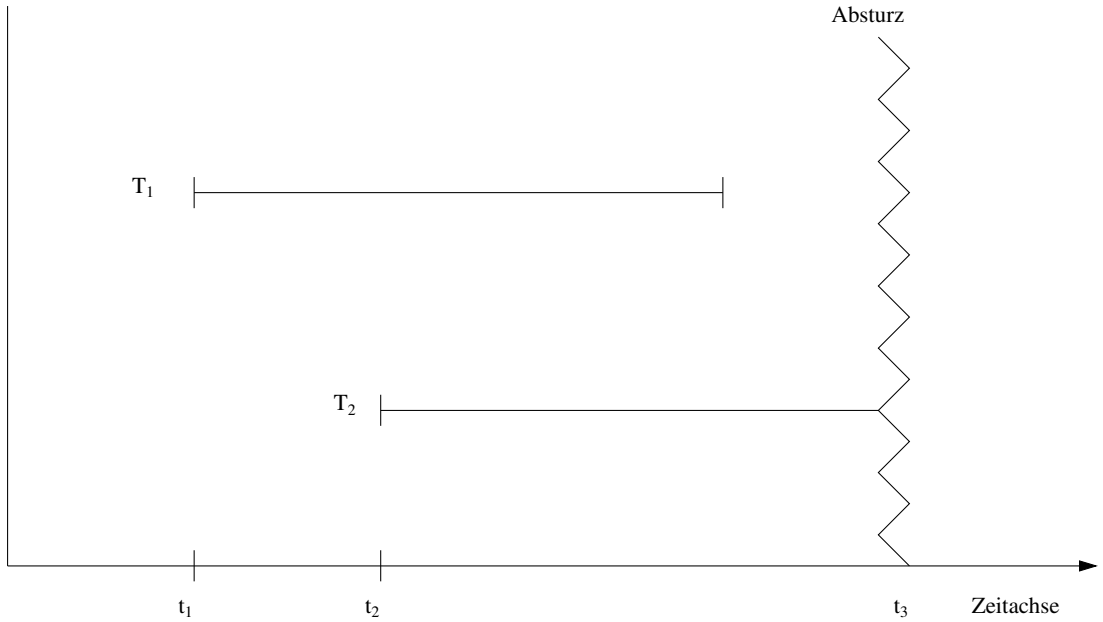
- *PageID*
  - die Kennung der Seite, auf der die Änderungsoperation vollzogen wurde
  - Wenn eine Änderung mehr als eine Seite betrifft, müssen entsprechend viele Log-Einträge generiert werden
- *PrevLSN*,
  - Zeiger auf den vorhergehenden Log-Eintrag der jeweiligen Transaktion
  - Diesen Eintrag benötigt man aus Effizienzgründen

Das WAL-Prinzip

Write Ahead Log-Prinzip

1. Bevor eine Transaktion festgeschrieben (**committed**) wird, müssen alle „zu ihr gehörenden“ Log-Einträge ausgeschrieben werden.
2. Bevor eine modifizierte Seite ausgelagert werden darf, müssen alle Log-Einträge, die zu dieser Seite gehören, in das temporäre und das Log-Archiv ausgeschrieben werden.

Wiederanlauf nach Fehler

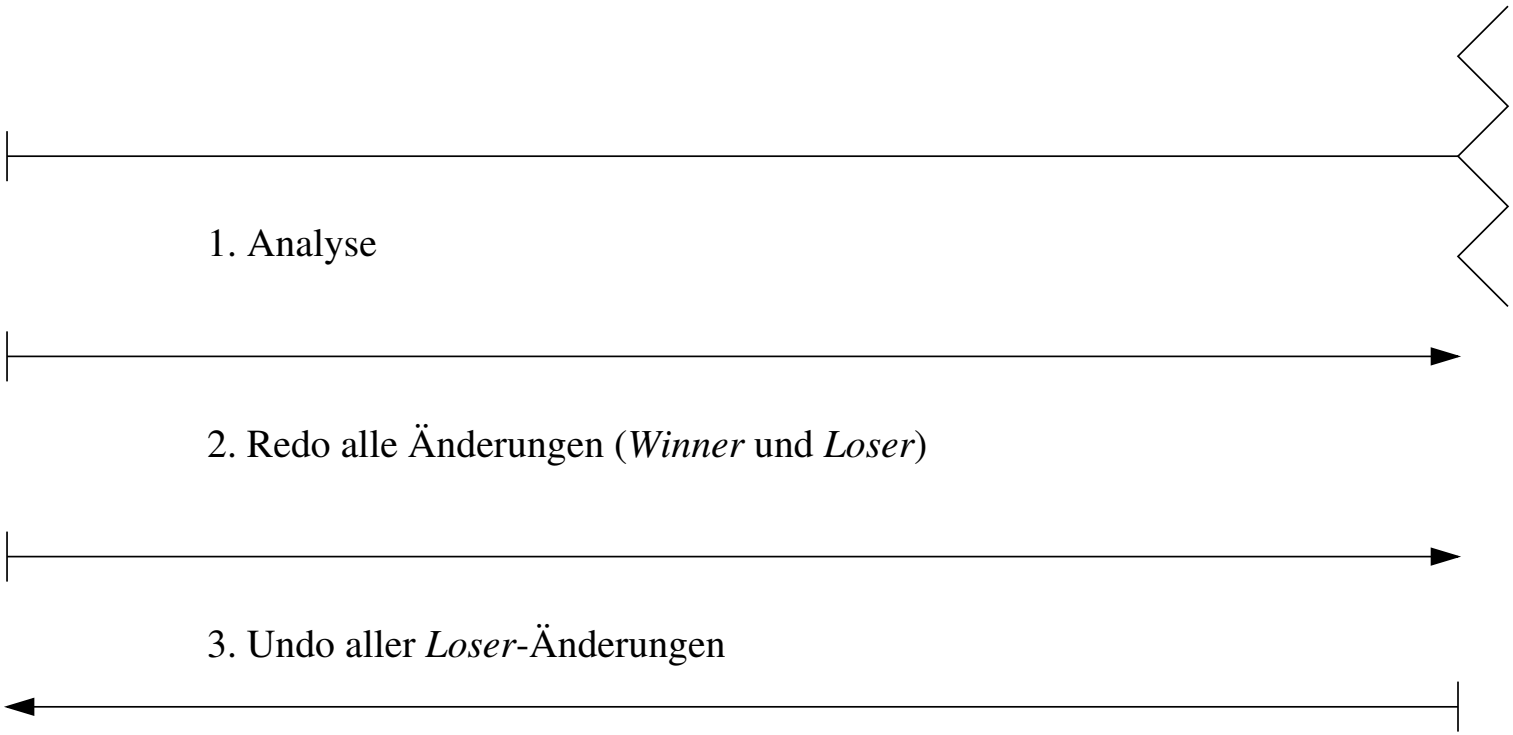


- TAs der Art T_1 sind *Winner*: müssen vollständig nachvollzogen werden
- TAs der Art T_2 sind *Loser*: müssen rückgängig gemacht werden

Phasen des Wiederanlaufs

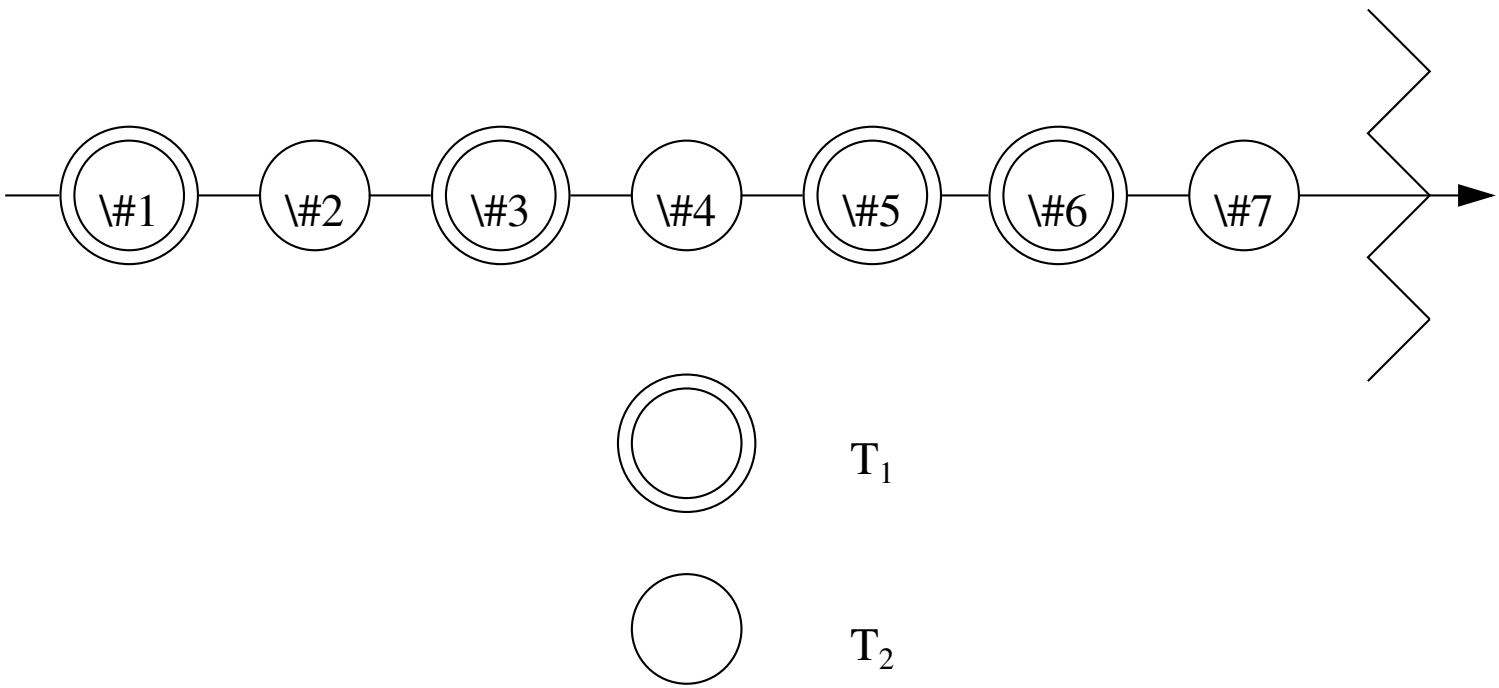
- *Analyse*:
  - Ermittlung der *Winner*-Menge von Transaktionen des Typs T_1
  - Ermittlung der *Loser*-Menge von Transaktionen der Art T_2.
- *Wiederholung der Historie*:
  - *Alle* protokollierten Änderungen werden in der Reihenfolge ihrer Ausführung in die Datenbasis eingebracht
- *Undo der Loser*:
  - Die Änderungoperationen der *Loser*-Transaktionen werden in umgekehrter Reihenfoge ihrer ursprünglichen Ausführung rückgängig gemacht

Phasen des Wiederanlaufs(2)

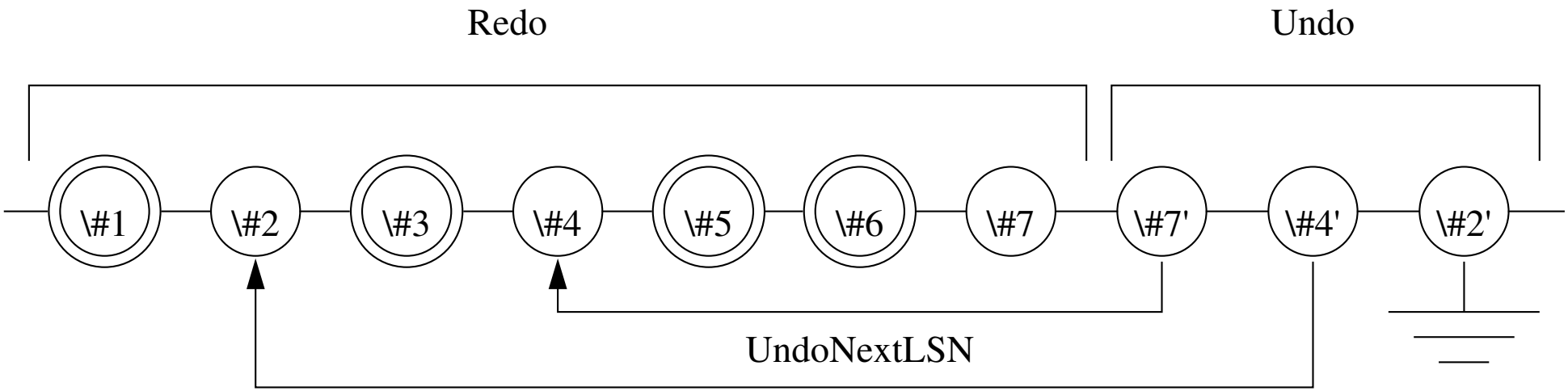


Fehlertoleranz Wiederanlauf

$$undo(undo(\cdots(undo(a))\cdots)) = undo(a)$$
$$redo(redo(\cdots(redo(a))\cdots)) = redo(a)$$



Fehlertoleranz Wiederanlauf(2)



- Kompensationseinträge (CLR: compensating log record) für rückgängig gemachte Änderungen.
- #7 ist CLR für #7
- #4 ist CLR für #4

Logeinträge nach abgeschlossenem Wiederanlauf II

- CLRs sind durch spitze Klammern <...> gekennzeichnet.
- der Aufbau eines CLR ist wie folgt
  - LSN
  - TA-Identifikator
  - betroffene Seite
  - Redo-Information
  - PrevLSN
  - UndoNxtLSN (Verweis auf die nächste rückgängig zu machende Änderung)
- CLRs enthalten keine Undo-Information
  - warum nicht?