Transaktionsverwaltung

- Commit
- Eigenschaften von Transaktionen (ACID)
- Transaktionen in SQL

Kapitel 9

Transaktionsverwaltung

Beispiel einer typischen Transaktion in einer Bankanwendung:

1. Lese den Kontostand von *A* in die Variable *a*: **read**(*A*,*a*);
2. Reduziere den Kontostand um 50.- DM: *a* := *a* – 50;
3. Schreibe den neuen Kontostand in die Datenbasis: **write**(*A*,*a*);
4. Lese den Kontostand von *B* in die Variable *b*: **read**(*B*,*b*);
5. Erhöhe den Kontostand um 50,- DM: *b* := *b* + 50;
6. Schreibe den neuen Kontostand in die Datenbasis: **write**(*B*, *b*);

Ein Systemabsturz zwischen 3. und 4. macht den Kunden unglücklich.

Die Gehaltszahlung eingeschoben zwischen 5. und 6. auch.

Operationen auf Transaktions-Ebene

In den klassischen Transaktionssystemen:

- **begin of transaction (BOT)**: Mit diesem Befehl wird der Beginn einer eine Transaktion darstellende Befehlsfolge gekennzeichnet.
- **commit**: Hierdurch wird die Beendigung der Transaktion eingeleitet. Alle Änderungen der Datenbasis werden durch diesen Befehl festgeschrieben, d.h. sie werden dauerhaft in die Datenbank eingebaut.
- **abort**: Dieser Befehl führt zu einem Selbstabbruch der Transaktion. Das Datenbanksystem muss sicherstellen, dass die Datenbasis wieder in den Zustand zurückgesetzt wird, der vor Beginn der Transaktionsausführung existierte.

Erweiterte Operationen auf Transaktions-Ebene

Zusätzlich in neuen Datenbankanwendungen:

- **define savepoint**
  - Definiert einen Sicherungspunkt, auf den sich die (noch aktive) Transaktion zurücksetzen lässt.
  - Das DBMS muss sich alle bis zu diesem Zeitpunkt ausgeführten Änderungen an der Datenbasis „merken“.
  - Diese Änderungen dürfen aber noch nicht in der Datenbasis festgeschrieben werden, da die Transaktion durch ein **abort** immer noch gänzlich aufgegeben werden kann.

Erweiterte Operationen auf Transaktions-Ebene

Zusätzlich in neuen Datenbankanwendungen (Forts.):

- **backup transaction**
  - Dient dazu, die noch aktive Transaktion auf den jüngsten – also den zuletzt angelegten – Sicherungspunkt zurückzusetzen.
  - Abhängig von der Funktionalität des Systems ist auch ein Rücksetzen auf weiter zurückliegende Sicherungspunkte möglich.

Um diese Funktionalität zu realisieren, benötigt man selbstverständlich entsprechend mehr Speicherkapazität, um die Zustände mehrerer Sicherungspunkte temporär abzuspeichern – oder wie wir in Kapitel 10 sehen werden, mehr Zeit, um die ausgeführten Operationen rückgängig zu machen.

Abschluss einer Transaktion

Für den Abschluss einer Transaktion gibt es zwei Möglichkeiten:

1. Den erfolgreichen Abschluss durch ein **commit**.

2. Den erfolglosen Abschluss durch ein **abort**.

Eigenschaften von Transaktionen: ACID

- **A**tomicity (Atomarität)
  - Transaktionen müssen komplett durchgeführt werden: Alles oder nichts
  - Bsp.: Umbuchung von einem Konto auf ein anderes. Abbruch zwischen Ab- und Zubuchung nicht erlaubt.
- **C**onsistency
  - Ein konsistenter DB-Zustand muss in einen konsistenten Zustand übergehen
  - Bsp.: doppelte Buchführung: Saldo muss gleich null bleiben
- **I**solation
  - Jede Transaktion hat die DB „für sich allein“
  - Bsp.: Der letzte freie Sitzplatz im Flugzeug darf nicht parallel vergeben werden.
- **D**urability (Dauerhaftigkeit)
  - Änderungen erfolgreicher Transaktionen dürfen nie verloren gehen
  - Bsp.: Scheine an der Uni ;-)

Eigenschaften von Transaktionen

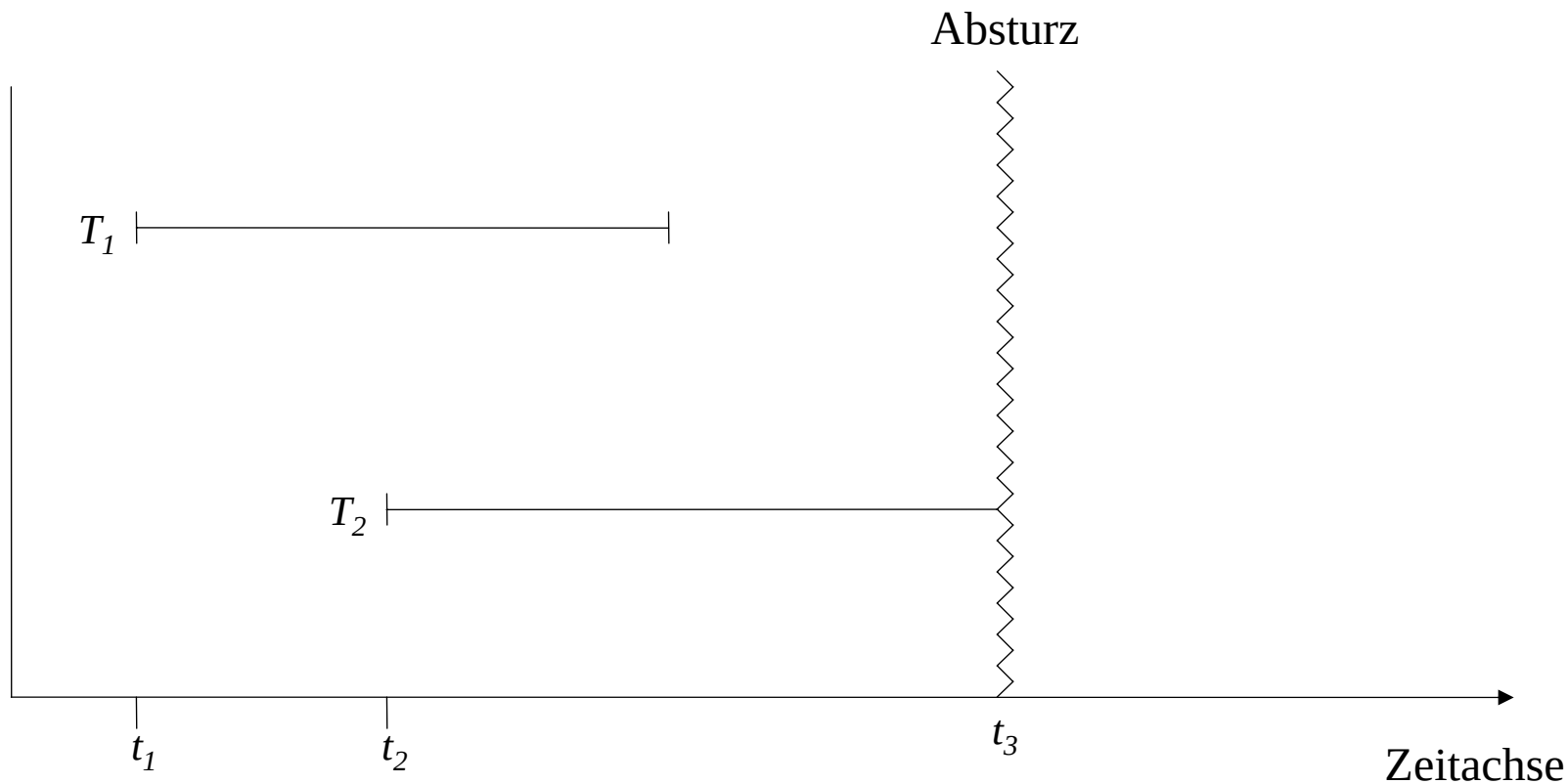

Abb.: Transaktionsbeginn und –ende relativ zu einem Systemabsturz

Transaktionsverwaltung in SQL

commit work

- Die in der Transaktion vollzogenen Änderungen werden – falls keine Konsistenzverletzung oder andere Probleme aufgedeckt werden – festgeschrieben.
- Das Schlüsselwort **work** ist optional, d.h. das Transaktionsende kann auch einfach mit **commit** „*befohlen*“ werden.

rollback work

- Alle Änderungen sollen zurückgesetzt werden.
- Anders als der **commit**-Befehl muss das DBMS die „*erfolgreiche*“ Ausführung eines rollback-Befehls immer garantieren können.

Transaktionsverwaltung in SQL

Beispielsequenz auf Basis des Universitätsschemas:

```
insert into Vorlesungen
        values (5275, `Kernphysik`, 3, 2141);
insert into Professoren
        values (2141, `Meitner`, `C4`, 205);
commit work
```

Zustandsübergangs-Diagramm für Transaktionen