Multi-User-Applikationen objektorientiert realisieren

25.11.24

Transaktionen in DBMS

Handlungsnotwendige Kenntnisse

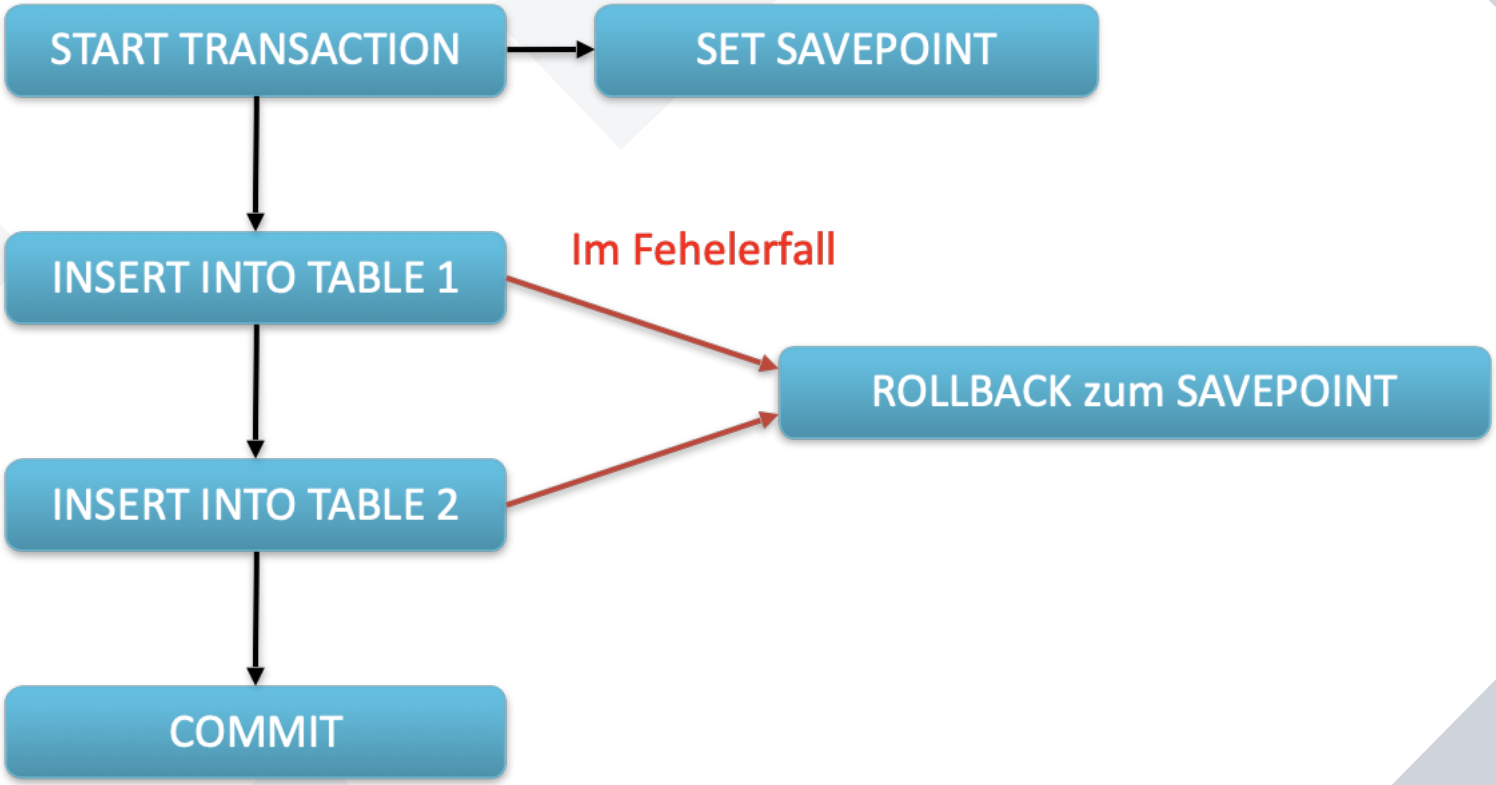
- 1.1 Kennt Anforderungen an das Datenbankmanagement-System bezüglich Multi-User-Fähigkeit.
- 3.3 Kennt Möglichkeiten, um Transaktionen im DBMS sicherzustellen.

Ziele

- Du kannst die verschiedenen Isolationsprobleme auf Datenbank Ebene erklären
- Du kennst die verschiedenen Isolationslevel
- Du kannst die vier ACID Kriterien aufzählen

«Als Transaktion bezeichnet man in der Informatik eine Folge von Programmschritten, die als eine logische Einheit betrachtet werden, weil sie den Datenbestand nach fehlerfreier und vollständiger Ausführung in einem konsistenten Zustand hinterlassen. Daher wird für eine Transaktion insbesondere gefordert, dass sie entweder vollständig und fehlerfrei oder gar nicht ausgeführt wird.» Wikipedia

Transaktion Ablauf



Refresher MSSQL Statements

- **BEGIN TRANSACTION**
  - Neue Transaktion starten
- **COMMIT TRANSACTION**
  - Änderungen persistieren
- **ROLLBACK TRANSACTION**
  - Änderungen verwerfen

Beispiel Transaktion

```
BEGIN TRANSACTION;

UPDATE TICKET SET AMOUNT = 2 WHERE ID = 1;

UPDATE TICKET SET AMOUNT = 3 WHERE ID = 4;

COMMIT TRANSACTION;
```

ACID-Prinzip

- Eigenschaften von verlässlichen Transaktionen
  - Atomarität
  - Konsistenz
  - Isolation
  - Dauerhaftigkeit

ACID-Prinzip

- Transaktionen werden ganz oder gar nicht ausgeführt
- Transaktionen können aus mehreren SQL-Statements bestehen
- Bei Abbruch der Transaktion (Rollback) bleibt die Datenbank unverändert

Atomicity

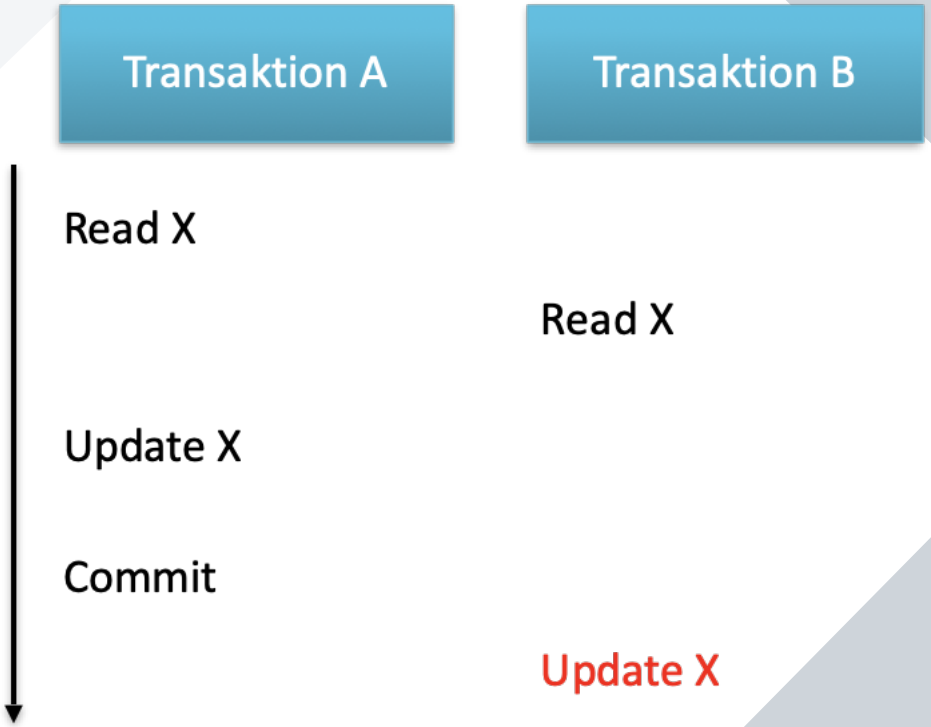
ACID-Prinzip

- Änderungen der Transaktion werden dauerhaft gespeichert
- Dauerhafte Speicherung darf nicht durch einen Systemfehler verhindert werden

Durability

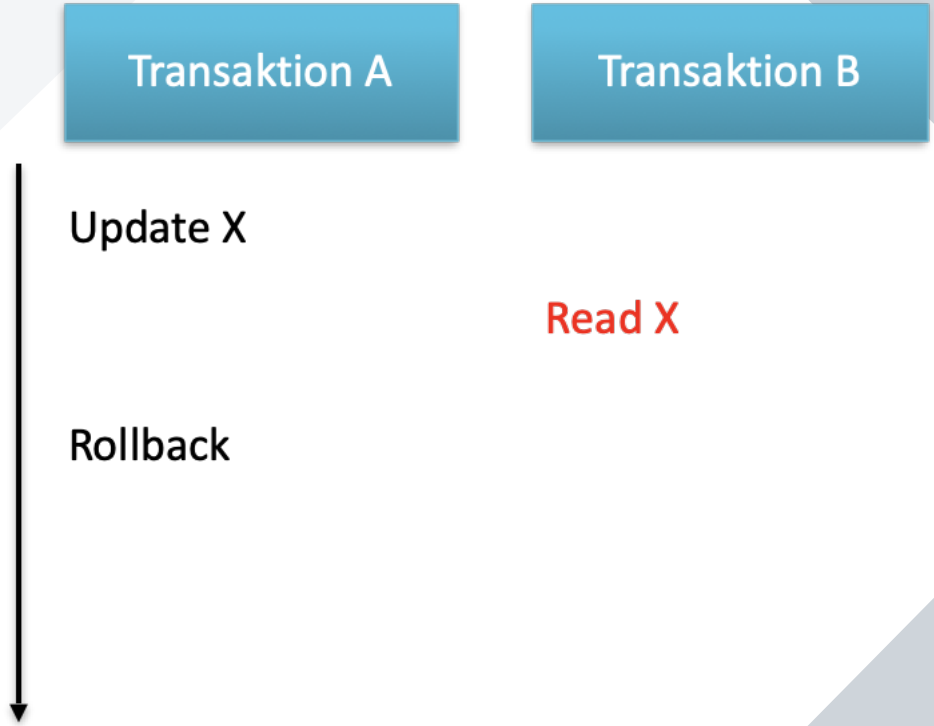
Isolationsprobleme – Lost Update

- Zwei Transaktionen modifizieren parallel den gleichen Datensatz
- Nur die Änderung einer Transaktion wird übernommen



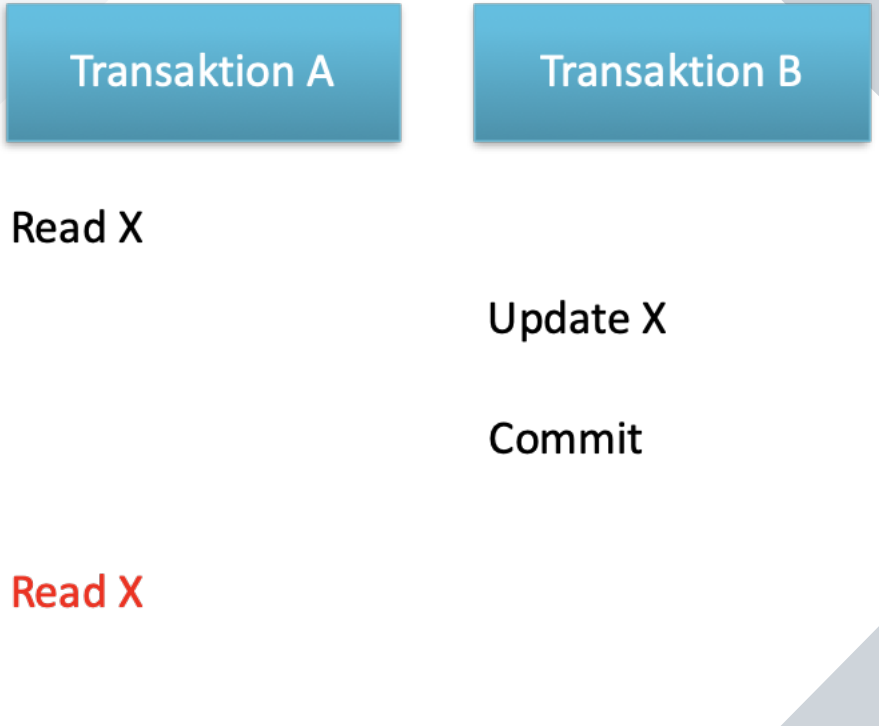
Isolationsprobleme – Dirty Read

- Daten einer nicht abgeschlossenen Transaktion werden von einer anderen Transaktion gelesen



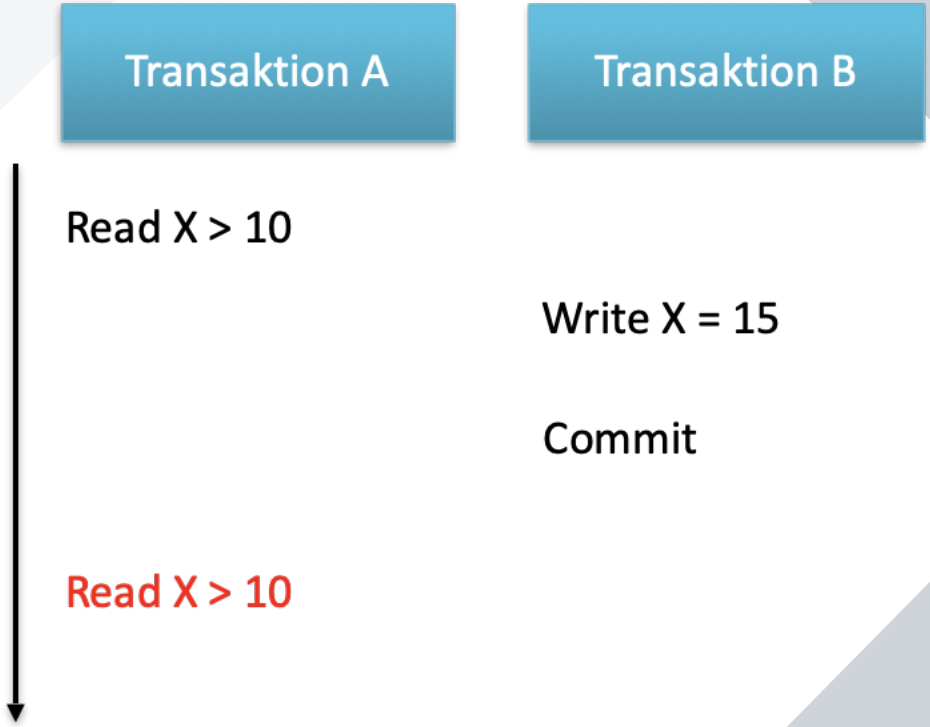
Isolationsprobleme – Non-repeatable Read

- Wiederholte Lesevorgänge liefern unterschiedliche Ergebnisse
- Die Anzahl Datensätze bleibt bei den beiden Lesevorgängen gleich, der Inhalt ändert sicher aber



Isolationsprobleme – Phantom Read

- Suchkriterien treffen während einer Transaktion auf unterschiedliche Datensätze zu
- Die Anzahl Resultate bei den beiden Lesevorgängen ist unterschiedlich



Isolationslevel

- Read uncommitted / Dirty Read
  - Uncommittete Daten von anderen Transaktionen werden gelesen
  - Bei einem SELECT wird **kein** Lock gemacht
- Read committed
  - Nur committete Daten werden gelesen
  - Jedes SELECT Statement hat einen eigenen Snapshot

Isolationslevel

- Repeatable Read
  - Ein SELECT Statement gibt immer die gleichen Daten / Snapshot zurück
  - Default bei InnoDB (MySQL)
- Serializable
  - Transaktionen werden so ausgeführt, als wären sie nacheinander
  - Nicht ideal aus Concurrency Sicht

Isolationslevel (generell)

Level	Write Lock	Read Lock	Range Lock
Read uncommitted	-	-	-
Read committed	X	Shared Lock	-
Repeatable read	X	X	-
Serializable	X	X	X

Isolationslevel (InnoDB)

Level	Lost Updates	Dirty Read	Non-repeatable Read	Phantom Read
Read uncommitted	unmöglich	möglich	möglich	möglich
Read committed	unmöglich	unmöglich	möglich	möglich
Repeatable Read	unmöglich	unmöglich	unmöglich	möglich
Serializable	unmöglich	unmöglich	unmöglich	unmöglich

Fragen?