

Tag 4

Inhaltsverzeichnis

- **Normalformen**
 - Problem
 - Formen (1-4)
 - Weitere Formen
- **Transaktionen**
 - Synchronisationsprobleme
 - Überblick
 - Autocommit
 - Locking
 - Savepoints
 - Isolation levels
- **Übungen**

Normalformen

Problematik

BuchTitel	ISBN	Autor1	Autor2	Verlag
Datenbanksysteme	123	Kemper	Eickler	Oldenbourg
MySQL5	456	Kofler	"null"	Addison-Wesley
Linux	789	Kofler	"null"	Addison-Wesley

Anomalien

- Einfügung
- Update
- Löschen

Zum Beispiel

- Einen dritten Autor beim DB-Buch addieren
- Autor Kofler oder Verlag AW anpassen

Normalformen

1_NF, vorher

- Jeder Datensatz mit **Primärschlüssel** identifizierbar
- Jedes Attribut der Relation muss einen **atomaren** Wertebereich haben, und die Relation muss **frei von Wiederholungsgruppen** sein.

BuchTitel	ISBN	Autor	Verlag
Datenbanksysteme	123	Kemper, Eickler	Oldenbourg
MySQL5	456	Kofler	Addison-Wesley
Linux	789	Kofler	Addison-Wesley

Normalformen

1_NF, Zwischenresultat

- Jeder Datensatz mit **Primärschlüssel** identifizierbar
- Jedes Attribut der Relation muss einen **atomaren** Wertebereich haben, und die Relation muss **frei von Wiederholungsgruppen** sein.



BuchKey	BuchTitel	ISBN	Autor	Verlag
0	Datenbanksysteme	123	Kemper	Oldenbourg
1	MySQL5	456	Kofler	Addison-Wesley
2	Linux	789	Kofler	Addison-Wesley
3	Datenbanksysteme	123	Eickler	Oldenbourg



Redundanz eingeführt... => es braucht mehr Tabellen!

Normalformen

1_NF, **nachher**

- Jeder Datensatz mit **Primärschlüssel** identifizierbar
- Jedes Attribut der Relation muss einen **atomaren** Wertebereich haben, und die Relation muss **frei von Wiederholungsgruppen** sein.

ID	Titel	ISBN	VerlagID
0	Linux	123	1
1	MySQL 5	4567	2
2	Datenbanksysteme	9876	3

ID	Verlag
1	Addison-Wesley
2	PrenticeHall
3	Hanser

ID	Vorname	Name
1	Michael	Koffler
2	Alfons	Kemper
3	André	Eickler

AutorID	BuchID
1	0
1	1
2	3
3	3

Normalformen

2_NF, vorher

- 1_NF
- Wenn jedes Nichtschlüsselattribut von jedem Schlüsselkandidaten **voll funktional** abhängig ist.
<=> Jedes nicht-primäre Attribut (nicht Teil eines Schlüssels) ist **vom gesamten** Schlüssel abhängig, nicht nur von einem Teil davon

Beispiel [Bearbeiten]

CD_Lieder

CD_ID	Albumtitel	Interpret	Jahr der Gründung	Track	Titel
4711	Not That Kind	Anastacia	1999	1	Not That Kind

1_NF Tabellen mit nicht zusammengesetzten Schlüsseln sind *automatisch* in 2_NF

Normalformen

2_NF, **nachher**

- 1_NF
- Wenn jedes Nichtschlüsselattribut von jedem Schlüsselkandidaten **voll funktional** abhängig ist.
<=> Jedes nicht-primäre Attribut (nicht Teil eines Schlüssels) ist **vom gesamten** Schlüssel abhängig, nicht nur von einem Teil davon

CD				Lieder		
CD_ID	Albumtitel	Interpret	Jahr der Gründung	CD_ID	Track	Titel
4711	Not That Kind	Anastacia	1999	4711	1	Not That Kind

Hier hat man die Tabelle CD_Lieder in zwei Tabellen *zerlegt*

Normalformen

3_NF, vorher

- 2_NF
- Kein "Nichtschlüssel" Attribut hängt von irgendeinem Schlüsselkandidaten **transitiv** ab.

Ein Attribut A ist vom Schlüsselkandidaten C *transitiv* abhängig, wenn es ein Attribut B gibt, so dass $(C \rightarrow B)$ und $(B \rightarrow A)$.

CD

CD_ID	Albumtitel	Interpret	Jahr der Gründung
4711	Not That Kind	Anastacia	1999
4713	Freak of Nature	Anastacia	1999
4712	Wish You Were Here	Pink Floyd	1964

Normalformen

3_NF, **nachher**

- 2_NF
- Kein "Nichtschlüssel" Attribut hängt von irgendeinem Schlüsselkandidaten **transitiv** ab.

Ein Attribut A ist vom Schlüsselkandidaten C *transitiv* abhängig, wenn es ein Attribut B gibt, so dass $(C \rightarrow B)$ und $(B \rightarrow A)$.

CD

CD_ID	Albumtitel	Interpret
4711	Not That Kind	Anastacia
4713	Freak of Nature	Anastacia
4712	Wish You Were Here	Pink Floyd

Künstler

Interpret	Jahr der Gründung
Anastacia	1999
Pink Floyd	1964

Hier hat man die Tabelle CD in zwei Tabellen zerlegt

Normalformen

4_NF, vorher

- 3_NF
- Vierte Normalform (4_NF)
(es darf nicht **mehrere, voneinander unabhängige, 1:n-Beziehungen** in einer Relation geben)

Besitz

Personnummer	Haustier	Fahrzeug
1	Katze	Volkswagen
1	Katze	Ferrari
1	Pelikan	Volkswagen
1	Pelikan	Ferrari
2	Hund	Porsche

Normalformen

4_NF, **nachher**

- 3_NF
- Vierte Normalform (4_NF)
(es darf nicht **mehrere, voneinander unabhängige, 1:n-Beziehungen** in einer Relation geben)

Haustier

Personnummer	Haustier
1	Katze
1	Pelikan
2	Hund

Fahrzeug

Personnummer	Fahrzeug
1	Volkswagen
1	Ferrari
2	Porsche

Normalformen

Weitere Formen

- 1_NF
- 2_NF
- 3_NF
- *Boyce-Codd-Normalform (BC_NF)*
- 4_NF
- *5_NF*
- *6_NF*

Transaktionen

Definition

- Reihenfolge von **zusammengehörigen** Operationen
- Für Datenbank, Kontoüberweisung, etc...
- Wechsel zwischen **konsistenten** Zuständen
- Muss **ACID**-Eigenschaften erfüllen (Definition folgt)

Transaktionen

Synchronisationsprobleme

- Verlorene Updates
- Schreib-Lese-Konflikt (Dirty Read)
- Nichtwiederholbares Lesen
- Phantomproblem

Transaktionen

Beispiel "Verlorene Updates"

Zeit

Programm 1

Programm 2

Programm 1 liest das Konto X

Programm 2 liest das Konto X

Programm 1 ändert Konto X und schreibt den neuen Stand

Programm 2 ändert Konto X und schreibt den neuen Stand

Die Aktualisierung von Programm 1 ist verloren gegangen

Transaktionen

Beispiel "Schreib-Lese-Konflikt"

Zeit

Programm 1

Programm 2

Programm 1 startet
eine Transaktion

Programm 1 fügt Zeile A ein

Programm 2 liest Zeile A

Programm 1 macht das Einfügen
von Zeile A rückgängig (Rollback)

Programm 2 hat mit A ein Problem

Transaktionen

Beispiel "Nichtwiederholbares Lesen"

Zeit

Programm 1

Programm 2

Programm 1 liest Konto X. Es ist
nicht leer...

Programm 2 liest Konto X

Programm 2 leert Konto X

Aufgrund des Zustands Konto X
erlaubt Programm 1 eine Transaktion;
Der Kontostand ist aber nicht
gleich wie vorher...

Transaktionen

Phantomproblem

Zeit

Ziel: Durchschnitt der Artikel im Lager berechnen

Programm 1

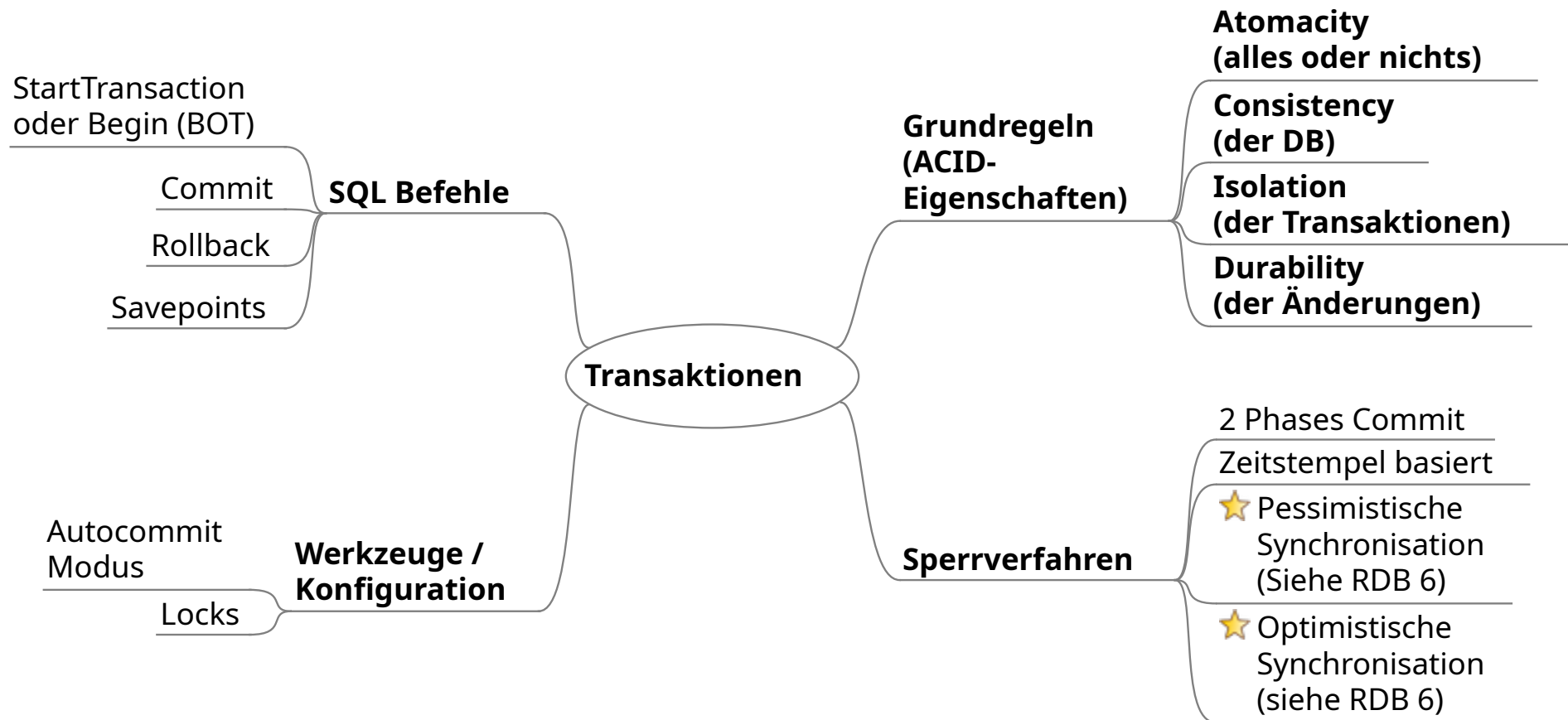
Programm 2

Programm 1 rechnet `sum(Artikel)`
des Lagers

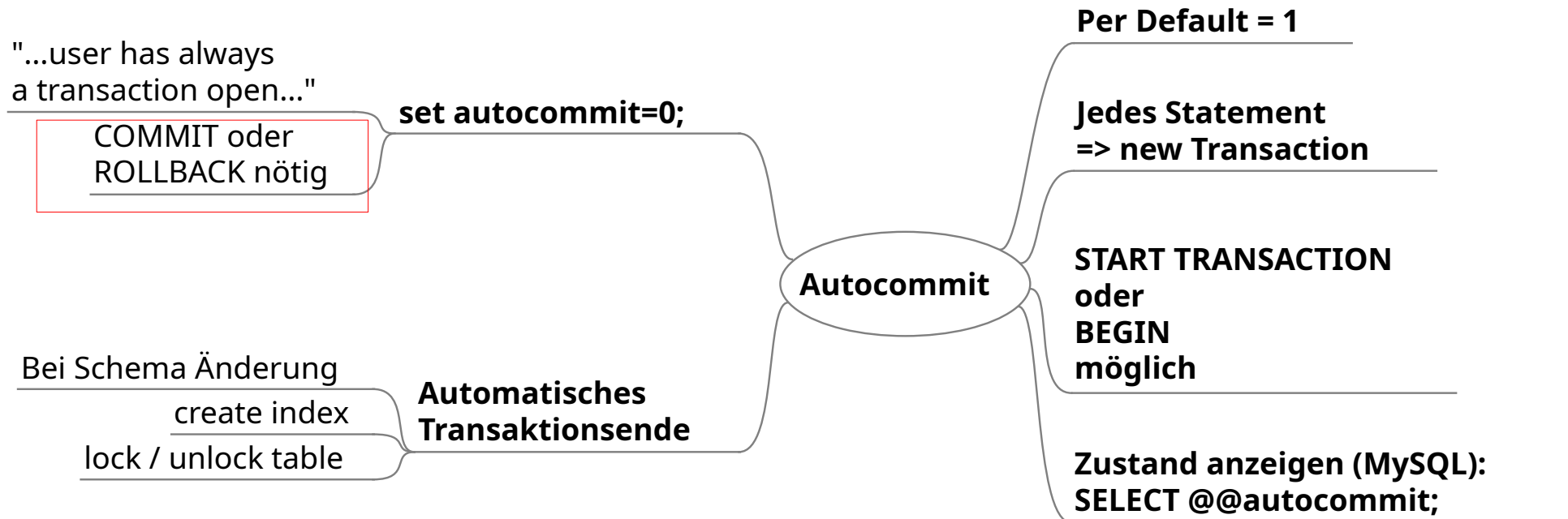
Programm 2 fügt neue Artikel
ins Lager

Programm 1 rechnet `count(Artikel)`
des Lagers.
Dieser Wert ist falsch...

Transaktionen Überblick



Transaktionen Autocommit



Transaktionen

Locking, Beispiel (1)

Zeit

Ohne Locking
(Guthaben Konto: 100€)

Person 1	Person 2
liest Kontostand (100€)	
	liest Kontostand (100€) Hebt 100€ ab schreibt neuen Kontostand (100€-100€ = 0€)
zahlt 50€ ein schreibt neuen Kontostand (100€+50€ = 150€)	

Neuer Stand: 150€ **falsch!**

Transaktionen

Locking, Beispiel (2)

Zeit

Mit Locking
(Guthaben Konto: 100€)

Person 1	Person 2
Zugriff auf Bankkonto wird gesperrt liest Kontostand (100€)	
	versucht Kontostand zu lesen Lock greift, Person 2 muss warten.
zahlt 50€ ein schreibt neuen Kontostand (100€+50€ = 150€) Zugriff auf Bankkonto wird freigegeben	
	Lock frei Zugriff auf Bankkonto wird gesperrt liest Kontostand (150€) hebt 100€ ab schreibt neuen Kontostand (150€-100€ = 50€) Zugriff auf Bankkonto wird freigegeben



B. Neuer Stand: 50€ **richtig!**

Fachhochschule

Gilles Maitre

RDB 4 - 22
Version 3.11

Transaktionen

Locking (READ)

- Nur für InnoDB-Tabellen und wenn autocommit == 0
- Kein *READ LOCK* Statement =>
SQL Select Erweiterung:

```
SELECT * FROM... WHERE konto='x' LOCK IN SHARE MODE;
```

- Lock auf **Datensatz-Ebene**
- Erlaubt anderen Transaktionen die ausgewählten Datensätze zu **lesen** aber **nicht zu schreiben**
- Gilt bis Ende der Transaktion

Transaktionen Locking (WRITE)

- Nur für InnoDB Tabellen und wenn autocommit == 0
- Kein *WRITE LOCK* =>
MySQL Select Erweiterung:

```
SELECT * FROM... WHERE konto='x' FOR UPDATE;
```

- Erlaubt anderen Transaktionen die ausgewählten Datensätze zu **lesen** (je nach IsolationLevel) aber **nicht zu schreiben**.
Transaktionen mit SELECT... LOCK IN SHARE MODE sind blockiert.
- Gilt bis Ende der Transaktion

Transaktionen

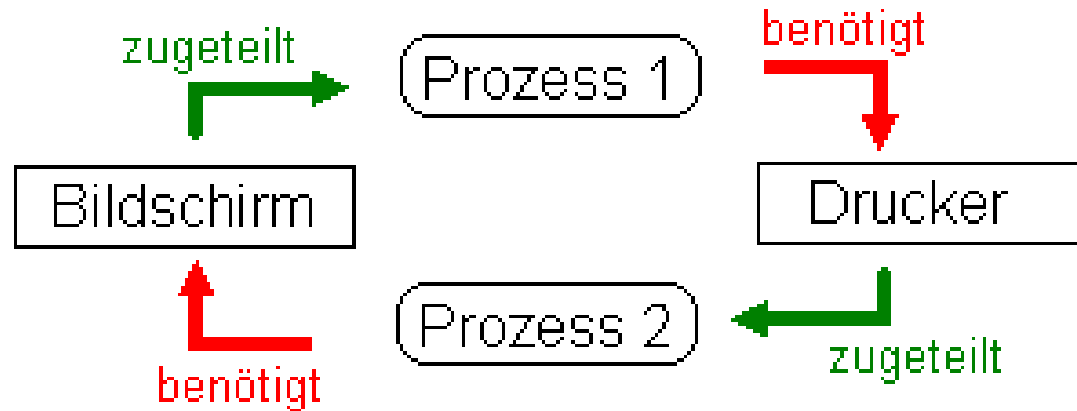
MySQL Locking für MyISAM

- Default: Kein Lock
- Lock auf **Tabellenebene**:

```
LOCK TABLE table_1 lockTyp_1, ...  
UNLOCK
```

- Lock Typen:
 - READ LOCAL
(Locks Updates, insert am Ende der Tabelle möglich)
 - READ
(Locks alle Arten von Updates)
 - WRITE
(Lock alle Lesen und Schreiben)

Transaktionen Locks and Deadlocks...



- InnoDB-Treiber erkennt es => im letzten Prozess
 - Fehler
 - Rollback der Transaktion
- "innodb_lock_wait_timeout=n", Default 50 Sek.

Transaktionen

Savepoints

```
START TRANSACTION;  
...  
SAVEPOINT point_1;  
...  
SAVEPOINT point_2;  
...  
<Exception>!  
=> "ROLLBACK TO SAVEPOINT point_1;"  
...  
COMMIT;
```

MySQL: Nur für InnoDB Tables und innerhalb der Transaktion

Transaktionen Isolation Levels

```
SET [session|global] TRANSACTION ISOLATION LEVEL
```

wenig isoliert



stark isoliert

```
READ UNCOMMITTED (=> "dirty reads" möglich)
|
READ COMMITTED (=> besser geschützt)
|
REPEATABLE READ (=> wie der Name sagt)
|
SERIALIZABLE (=> select ==
              select in share mode)
```

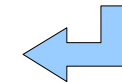
effizient



weniger effizient

```
mysql> SELECT @@transaction_ISOLATION, @@global.transaction_ISOLATION;
+-----+-----+
| @@tx_isolation | @@global.tx_isolation |
+-----+-----+
| REPEATABLE-READ | REPEATABLE-READ      |
+-----+-----+
```

MySQL Defaults



Übungen

1) Normalisieren Sie diese Datenbank mit den bekannten Normalformen.

Artikel: { Name, Typ, Herstellername, Herstelleradresse }

Lieferant: { ID, Name, Strasse, Stadt, Kanton, Land }

Lieferung: { ID, LieferantID, ArtikelName, Menge, PreisProArtikel }

2) Schreiben Sie eine SQL-Anweisung, die *nicht erfüllte Integritätsregeln* in der CD DB anzeigt (z.B. Unbekannte CD in "CDStueck").

Hinweis:

Fügen Sie zuerst eine unbekannte CD in "CDStueck" hinzu.

Dafür soll "SET foreign_key_checks=0" verwendet werden.