

Wirtschaftsinformatik 2 LE 08 – Transaktionen

Prof. Dr. Thomas Off

<http://www.ThomasOff.de/lehre/beuth/wi2>

Ziel

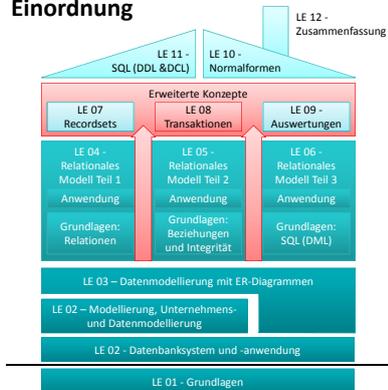
Ziel dieser Lehreinheit

- Konsistenzsicherung als Anforderung an Datenbanken wiederholen und vertiefen
- Konzepte der Transaktionen und deren Eigenschaften kennenlernen
- Anwendung von Transaktionen in SQL und mit MS Access
- Erläuterungen zum technischen Hintergrund der Transaktionsverarbeitung

Wirtschaftsinformatik 2 - LE 08 - Transaktionen

2

Einordnung



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

3

Inhalt

Ziel und Einordnung

Rückblick

Transaktionen

- Konsistenz und Integrität
 - Konsistenzsicherung als Ziel relationaler Datenbanken
 - Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung von Transaktionen
 - Anwendungsszenarien
 - Transaktionen in SQL
 - Transaktionen mit MS Access
- Technik der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 4

Rückblick

Wirtschaftsinformatik 2 - LE 06 - Recordsets 5

Rückblick (LE 07)

Recordset

- geordnete Menge von Datensätzen aus Tabelle(n) der Datenbank geladen
- Typen
 - Schnappschuss
 - Dynamische Verbindung
- Zeiger für Navigation über Datensätze (MoveNext, MoveFirst, ...)
- Zugriffsmöglichkeit auf Werte des Elementes, auf das Zeiger zeigt (Fields)
- Datensätze suchen (FindFirst, FindNext, ...)
- Modus für Ändern (Edit), Hinzufügen (AddNew) und anschließend Update oder alternativ Löschen (Delete)

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 6

Rückblick (LE 07)



Weitere VBA-Funktionen zum Datenzugriff

- Bisher: Zugriff auf mehrere Datensätze und deren Werte mittels Recordsets
- Jetzt: Ermittlung eines Ergebniswertes aus der Datenbank mittels Domänenfunktionen, z.B.
 - Ermitteln eines Wertes aus einem Datensatz,
 - Zählen aller Datensätze (anhand einer nicht leeren Spalte)

```
' Ausgabe  
Debug.Print DLookup("kndName", "tblKunden", "kndIdPk=2")  
Debug.Print DCount("kndIdPk", "tblKunden")
```

kndIdPk	kndName	kndVorname
1	Albers	Willi
2	Böhrs	Thomas
3	Dinkel	Ulrike
	(Neu)	

Direktbereich
Böhrs
3

Wirtschaftsinformatik 2 - LE 06 - Recordsets

7

Rückblick



LE 08 - ~~Transaktionen~~ **Transaktionen**



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

8

Rückblick (LE 02)



Definition "Datenbanksystem": Zusammenfassung und Bereitstellung konsistenter, integrierter und untereinander in Beziehung stehender Daten und Informationen über die Organisation dieser Daten zur Nutzung in mehreren Anwendungen



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

9

Rückblick (LE 02)  

Definition "Datenbanksystem": Zusammenfassung und Bereitstellung **konsistenter**, integrierter und untereinander in Beziehung stehender **Daten** und Informationen über die Organisation dieser Daten zur Nutzung in mehreren Anwendungen



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 10

Rückblick (LE 02)  

Anforderungen an Datenbanken in betrieblichen Anwendungen

– ...

- stellen sicher, dass die Daten korrekt sind
 - physisch korrekt gespeichert
 - logisch korrekt, so dass keine Widersprüche existieren
 - semantisch korrekt, so dass keine unsinnigen Daten gespeichert sind

– ...

➔ **Integrität der gespeicherten Daten ist eine Voraussetzung für die Konsistenz der Datenbank**

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 11

Inhalt 

Ziel und Einordnung

Rückblick

Transaktionen

- Konsistenz und Integrität
 - Konsistenzsicherung als Ziel relationaler Datenbanken
 - Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung von Transaktionen
 - Anwendungsszenarien
 - Transaktionen in SQL
 - Transaktionen mit MS Access
- Technik der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 12

Konsistenz und Integrität



Ziel relationaler Datenbanken ist Sicherung der Konsistenz/Integrität der gespeicherten Daten

- Konsistenz bezeichnet Korrektheit der gespeicherten Daten
 - physische Konsistenz: Daten werden technisch korrekt gespeichert und korrekt gelesen
 - logische Konsistenz: die zu speichernden Daten und Zusammenhänge sind für sich genommen korrekt
 - semantische Konsistenz: es werden keine unsinnigen Daten gespeichert
- Konsistenz ist nur gewährleistet, wenn die gespeicherten Daten alle Integritätsbedingungen erfüllen

Wirtschaftsinformatik 2 - LE 08 - Transaktionen

13

Konsistenz und Integrität



Ziel relationaler Datenbanken ist Sicherung der Konsistenz/Integrität der gespeicherten Daten

- wird erreicht für
 - Attribute/Spalten: durch Vorgabe und Prüfung des Wertebereichs
 - Entitäten/Relationen: durch Primärschlüssel und 1. Integritätsregel
 - Primärschlüssel muss eindeutig und darf niemals leer sein
 - Beziehungen/Fremdschlüssel: durch 2. Integritätsregel
 - Kein Fremdschlüssel (ungleich "leer"), dessen Wert im zugehörigen Primärschlüssel nicht existiert
 - Fachliche Zusammenhänge (z.B. Geburtsdatum von Kunden muss in der Vergangenheit liegen): durch Implementierung fachlicher Plausibilitätsregeln und Prüfung vor der Speicherung in der Datenbank

Wirtschaftsinformatik 2 - LE 08 - Transaktionen

14

Konsistenz und Integrität



Gefahren für die Konsistenz von Daten

- technische Fehler ausgelöst durch Rechnerabsturz, Stromausfall, Brand (Feuer, Löschwasser), ...
- logische und semantische Fehler durch konkurrierende Zugriffe mehrerer Benutzer



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

15

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

- Kunde Müller will Kunde Yilmaz 100 € überweisen

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 16

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

- Kunde Müller will Kunde Yilmaz 100 € überweisen
- Konsistenz: Summe des Guthabens von Müller und Yilmaz ist vor und nach der Überweisung gleich

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 17

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

- Kunde Müller will Kunde Yilmaz 100 € überweisen
- Konsistenz: Summe des Guthabens von Müller und Yilmaz ist vor und nach der Überweisung gleich

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 18

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- Kontostand von Müller wird um 100 € reduziert
- Kontostand von Yilmaz wird um 100 € erhöht

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 19

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- Kontostand von Müller wird um 100 € reduziert
- Kontostand von Yilmaz wird um 100 € erhöht

– SQL

```
UPDATE Konten
SET Saldo = Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo = Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 20

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- Kontostand von Müller wird um 100 € reduziert
- Kontostand von Yilmaz wird um 100 € erhöht

– SQL

```
UPDATE Konten
SET Saldo = Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo = Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

...

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 21

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- **Kontostand von Müller wird um 100 € reduziert**
- **Kontostand von Yilmaz wird um 100 € erhöht**

– SQL

```
UPDATE Konten
SET Saldo = Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo = Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 22

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- **Kontostand von Müller wird um 100 € reduziert**
- **Kontostand von Yilmaz wird um 100 € erhöht**

– SQL

```
UPDATE Konten
SET Saldo = Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo = Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 23

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Ablauf

- **Kontostand von Müller wird um 100 € reduziert**
- **Kontostand von Yilmaz wird um 100 € erhöht**

– SQL

```
UPDATE Konten
SET Saldo = Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo = Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €




Wirtschaftsinformatik 2 - LE 08 - Transaktionen 24

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Angenommen technischer Fehler (Rechnerabsturz, Stromausfall, Feuer, Löschwasser, ...) tritt zwischenzeitlich ein



Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500
Summe:			800 € + 500 € = 1300 €

↓

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500
Summe:			700 € + 500 € = 1200 €

↓

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600
Summe:			700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 25

Konsistenz und Integrität

Beispiel 1: Überweisung zwischen Konten einer Bank

– Angenommen technischer Fehler (Rechnerabsturz, Stromausfall, Feuer, Löschwasser, ...) tritt zwischenzeitlich ein

– ... dann ist die Datenbank nicht mehr konsistent (es fehlen 100 EUR)

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500
Summe:			800 € + 500 € = 1300 €

↓

Nachher?

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500
Summe:			700 € + 500 € = 1200 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 26

Konsistenz und Integrität

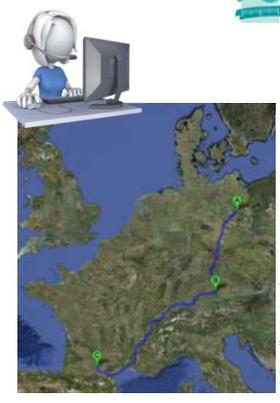
Beispiel 2: Flugbuchung TXL nach TLS via MUC




Wirtschaftsinformatik 2 - LE 08 - Transaktionen 27

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

- Konsistenz bedeutet
 - Vorher: es ist kein Flug gebucht
 - Nachher: es ist der gesamte Flug, bestehen aus einem Flug pro Teilstrecke gebucht
 - TXL → MUC
 - MUC → TLS



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 28

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

- Ablauf?



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 29

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

- Ablauf
 - Lesen, ob freie Plätze
 - Buchen, wenn auf beiden Teilstrecken ein Platz frei ist



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 30

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und

• Buchen, wenn auf beiden Teilstrecken ein Platz frei ist

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 31

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und
 - auf Teilstrecke MUC nach TLS verfügbar ist

• Buchen, wenn auf beiden Teilstrecken ein Platz frei ist

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 32

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und
 - auf Teilstrecke MUC nach TLS verfügbar ist

• Buchen, wenn auf beiden Teilstrecken ein Platz frei ist

- Teilstrecke TXL-MUC buchen

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 33

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und
 - auf Teilstrecke MUC nach TLS verfügbar ist
- Buchen, wenn auf beiden Teilstrecken ein Platz frei ist
 - Teilstrecke TXL-MUC buchen



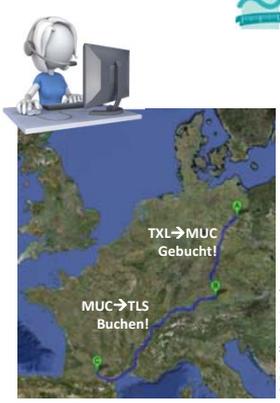
Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 34

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und
 - auf Teilstrecke MUC nach TLS verfügbar ist
- Buchen, wenn auf beiden Teilstrecken ein Platz frei ist
 - Teilstrecke TXL-MUC buchen
 - Teilstrecke MUC-TLS buchen



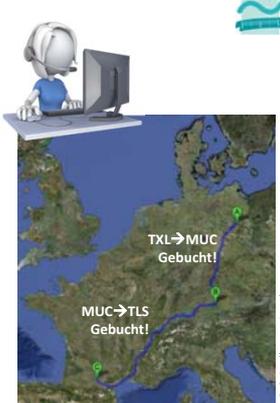
Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 35

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf

- Lesen, ob freie Plätze
 - auf Teilstrecke TXL nach MUC verfügbar ist und
 - auf Teilstrecke MUC nach TLS verfügbar ist
- Buchen, wenn auf beiden Teilstrecken ein Platz frei ist
 - Teilstrecke TXL-MUC buchen
 - Teilstrecke MUC-TLS buchen



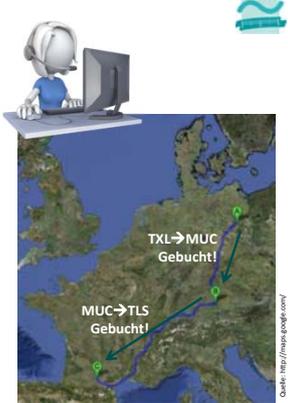
Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 36

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Konsistenter Zustand: es ist der gesamte Flug, bestehend aus einem Flug pro Teilstrecke gebucht

- TXL → MUC
- MUC → TLS



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 37

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 38

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 39

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze

A: TXL → MUC
Freie Plätze > 0?

B: TXL → MUC
Freie Plätze > 0?

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 40

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze

A: TXL → MUC
Freie Plätze > 0?

B: TXL → MUC
Freie Plätze > 0?

B: MUC → TLS
Freie Plätze > 0?

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 41

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze

A: TXL → MUC
Freie Plätze > 0?

B: TXL → MUC
Freie Plätze > 0?

A: MUC → TLS
Freie Plätze > 0?

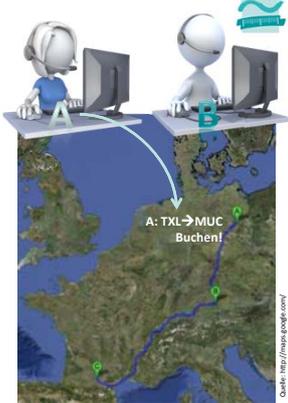
B: MUC → TLS
Freie Plätze > 0?

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 42

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
 – Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 43

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
 – Ablauf mit mehreren Benutzern

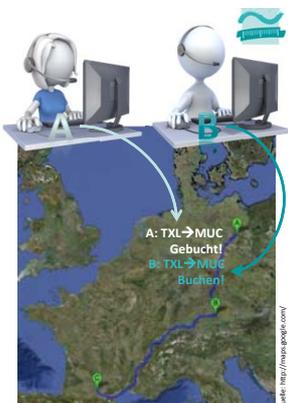
- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 44

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC
 – Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 45

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 46

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 2

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 47

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 2 → letzter Platz, puh... das war knapp

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 48

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 2 → letzter Platz, puh... das war knapp
- A: Buchen Teilstrecke 2

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 49

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

- A: Lesen freier Plätze
- B: Lesen freier Plätze
- A: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 1
- B: Buchen Teilstrecke 2 → letzter Platz, puh... das war knapp
- A: Buchen Teilstrecke 2 → Fehler, kein Platz mehr frei

Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 50

Konsistenz und Integrität
Beispiel 2: Flugbuchung TXL nach TLS via MUC

– Ablauf mit mehreren Benutzern

– Kann bei konkurrierenden Zugriffen dazu führen, dass Operationen nicht für alle Benutzer erfolgreich abgeschlossen werden

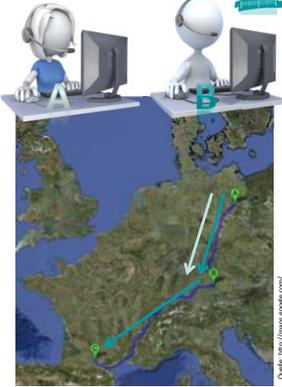
Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 51

Konsistenz und Integrität

Beispiel 2: Flugbuchung TXL nach TLS via MUC

- Ablauf mit mehreren Benutzern
- Kann bei konkurrierenden Zugriffen dazu führen, dass Operationen nicht für alle Benutzer erfolgreich abgeschlossen werden
- Konsistenzbedingung
 - nicht erfüllt für A
 - erfüllt für B



Quelle: <http://maps.google.com/>

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 52

Konsistenz und Integrität

Zwischenstand

- es gibt elementare Operationen
 - SELECT
 - INSERT
 - UPDATE
 - DELETE
- elementare Operationen werden immer vollständig oder gar nicht ausgeführt, d.h.
 - führen die Daten von einem technisch konsistenten Zustand in einen anderen technisch konsistenten Zustand
 - können logische Konsistenz nicht gewährleisten (z.B. Verschwinden von 100 € oder Buchung einer Teilstrecke anstelle der gesamten Reise)

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 53

Inhalt

Ziel und Einordnung

Rückblick

Transaktionen

- Konsistenz und Integrität
 - Konsistenzsicherung als Ziel relationaler Datenbanken
 - Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung von Transaktionen
 - Anwendungsszenarien
 - Transaktionen in SQL
 - Transaktionen mit MS Access
- Technik der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 54

Transaktionen

Definition: Folge von Datenbankoperationen,

- die hinsichtlich der Konsistenz/Integritätsbedingungen als atomare Einheit angesehen wird.¹
- die ausgehend von einem konsistenten Zustand der Datenbank immer in einen konsistenten Zustand führt.²
- die mit besonderen Kommandos
 - begonnen,
 - erfolgreich abgeschlossen oder
 - nicht erfolgreich beendet wird



1) vgl. A. Fink, G. Schneider, S. Voll: Grundlagen der Wirtschaftsinformatik, Physika-Verlag (Springer), 2002, S. 239
2) vgl. E. Schöcker: Datenbanken und SQL, Teubner Verlag, 1996, S. 59f.

Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit



Transaktionen

ACID-Eigenschaften

- Atomarität
 - Transaktion wird entweder vollständig oder gar nicht ausgeführt
 - tritt bei einer Operation der Transaktion ein Fehler auf, werden diese Operation und alle bereits (erfolgreich) ausgeführten Operationen zurückgesetzt
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit



Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
 - Transaktion führt die Datenbank stets von einem konsistenten Zustand in den nächsten konsistenten Zustand
 - Vor und nach der Ausführung der Transaktion sind stets alle Integritätsbedingungen erfüllt
- Isolation
- Dauerhaftigkeit



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 58

Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
 - Transaktionen laufen isoliert voneinander ab, d.h. aus Sicht des Benutzers verhält sich Datenbank so, als sei er der einzige Benutzer
 - parallele Transaktionen werden so ausgeführt, als würden sie nacheinander ablaufen, aber tatsächlich laufen sie parallel ab
 - DBMS stellt Isolation durch verschiedene Mechanismen sicher (z.B. aus Performance-Gründen nicht immer nacheinander sinnvoll)
- Dauerhaftigkeit



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 59

Transaktionen

ACID-Eigenschaften

- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit
 - abgeschlossene Transaktionen müssen auch nach einem unmittelbar anschließenden Fehlerzustand gespeichert sein
 - insbesondere, auch wenn
 - Stromausfall zum Löschen des Cache-Speichers im RAM führt
 - Festplattendefekt die Datenbank-Datei zerstört



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 60

Transaktionen

ACID-Eigenschaften

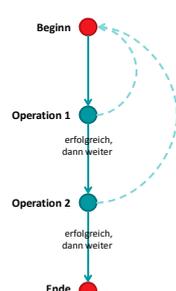
- Atomarität
- Konsistenz (Consistency)
- Isolation
- Dauerhaftigkeit



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 61

Transaktionen

Erfolgreich abgeschlossene Transaktion

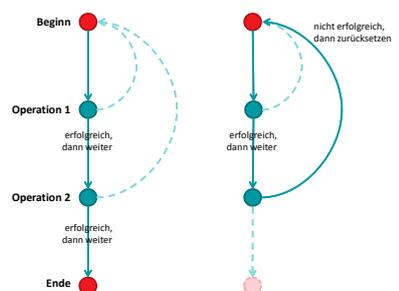


Wirtschaftsinformatik 2 - LE 08 - Transaktionen 62

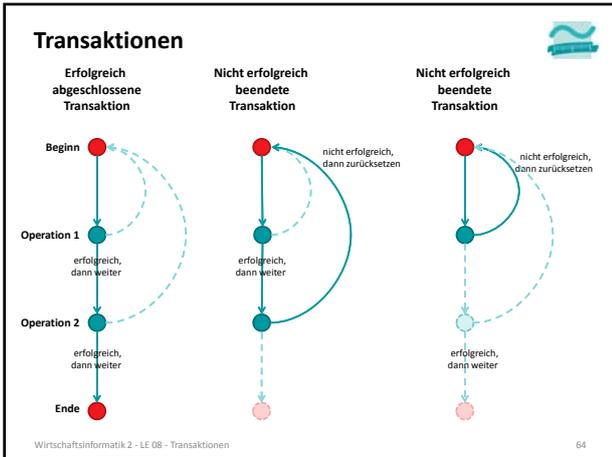
Transaktionen

Erfolgreich abgeschlossene Transaktion

Nicht erfolgreich beendete Transaktion



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 63



Inhalt

Ziel und Einordnung
Rückblick
Transaktionen

- Konsistenz und Integrität
 - Konsistenzsicherung als Ziel relationaler Datenbanken
 - Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung von Transaktionen
 - Anwendungsszenarien
 - Transaktionen in SQL
 - Transaktionen mit MS Access
- Technik der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 65

Anwendung von Transaktionen

Anwendung der Transaktionen auf

- Beispiel 1: Überweisung zwischen Konten
- Beispiel 2: Flugbuchung von TXL nach TLS via MUC

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 66

Anwendung von Transaktionen

Mit Transaktion:

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

↓

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 67

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
    saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo =
    saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 68

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
    Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo =
    saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 69

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich?

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 70

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich,
dann weiter

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 71

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich,
dann weiter

```
UPDATE Konten
SET Saldo =
  saldo + 100
WHERE KtoNr = 4567;
```

erfolgreich?

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 72

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich,
dann weiter

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 73

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich,
dann weiter

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Abschluss Transaktion

nicht erfolgreich,
dann zurücksetzen

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 74

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

Abschluss Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 75

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

Abschluss Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 76

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich?

Abschluss Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 77

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

erfolgreich,
dann weiter

Abschluss Transaktion

```
UPDATE Konten
SET Saldo =
  saldo + 100
WHERE KtoNr = 4567;
```

erfolgreich?

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 78

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Zurückrollen Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Diagramm: Ein Prozessfluss zeigt den Erfolg einer Transaktion. Ein roter Kreis markiert den Anfang. Ein blauer Kreis markiert den Erfolg. Ein grüner Kreis markiert den Zustand nach einer Unterbrechung. Ein roter Kreis markiert den Zustand nach einem Scheitern. Ein blauer Kreis markiert den Zustand nach dem Zurückrollen. Ein grüner Kreis markiert den Zustand nach dem Weitermachen.

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 79

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo =
  saldo + 100
WHERE KtoNr = 4567;
```

Zurückrollen Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Diagramm: Ein Prozessfluss zeigt den Erfolg einer Transaktion. Ein roter Kreis markiert den Anfang. Ein blauer Kreis markiert den Erfolg. Ein grüner Kreis markiert den Zustand nach einer Unterbrechung. Ein roter Kreis markiert den Zustand nach einem Scheitern. Ein blauer Kreis markiert den Zustand nach dem Zurückrollen. Ein grüner Kreis markiert den Zustand nach dem Weitermachen.

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 80

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;

UPDATE Konten
SET Saldo =
  saldo + 100
WHERE KtoNr = 4567;
```

Zurückrollen Transaktion

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Diagramm: Ein Prozessfluss zeigt den Erfolg einer Transaktion. Ein roter Kreis markiert den Anfang. Ein blauer Kreis markiert den Erfolg. Ein grüner Kreis markiert den Zustand nach einer Unterbrechung. Ein roter Kreis markiert den Zustand nach einem Scheitern. Ein blauer Kreis markiert den Zustand nach dem Zurückrollen. Ein grüner Kreis markiert den Zustand nach dem Weitermachen.

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 81

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

Abschluss Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 82

Anwendung von Transaktionen

Mit Transaktion:

Beginn Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo - 100
WHERE KtoNr = 2345;
```

Abschluss Transaktion

```
UPDATE Konten
SET Saldo =
  Saldo + 100
WHERE KtoNr = 4567;
```

Vorher:

Konten	KtoNr	Name	Saldo
	2345	Müller	800
	4567	Yilmaz	500

Summe: 800 € + 500 € = 1300 €

Zwischenzeitlich:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	500

Summe: 700 € + 500 € = 1200 €

Nacher:

Konten	KtoNr	Name	Saldo
	2345	Müller	700
	4567	Yilmaz	600

Summe: 700 € + 600 € = 1300 €

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 83

Anwendung von Transaktionen

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 84

Anwendung von Transaktionen

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 85

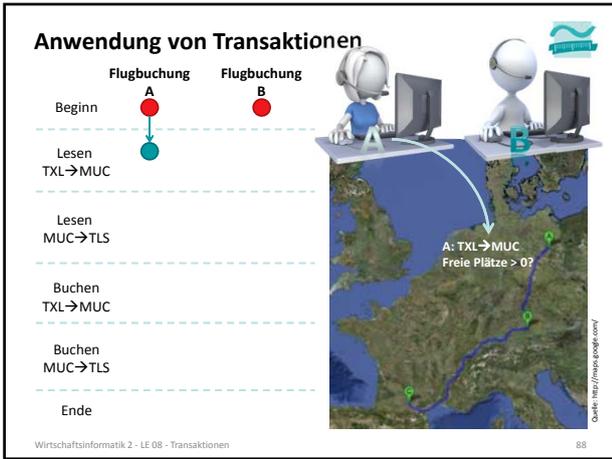
Anwendung von Transaktionen

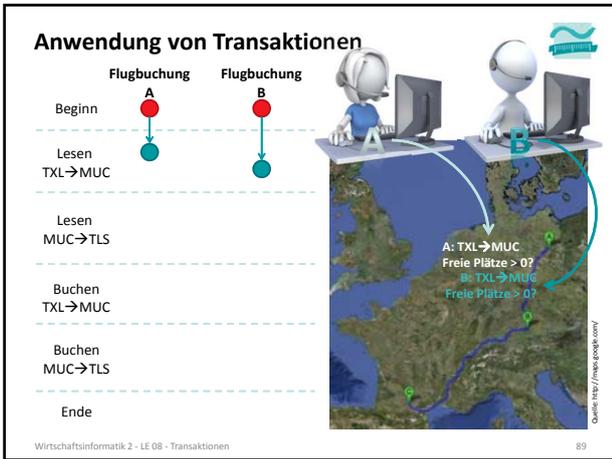
Wirtschaftsinformatik 2 - LE 08 - Transaktionen 86

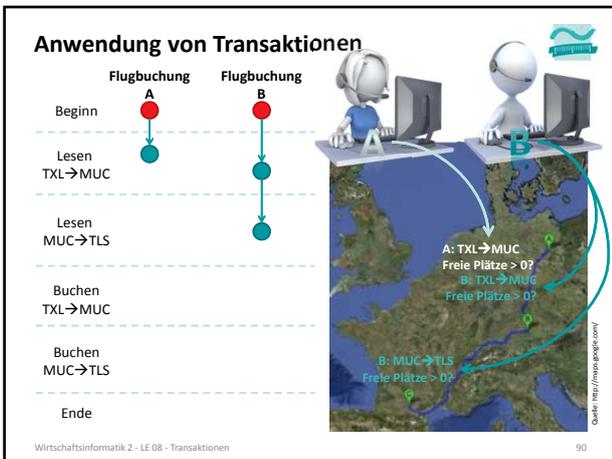
Anwendung von Transaktionen

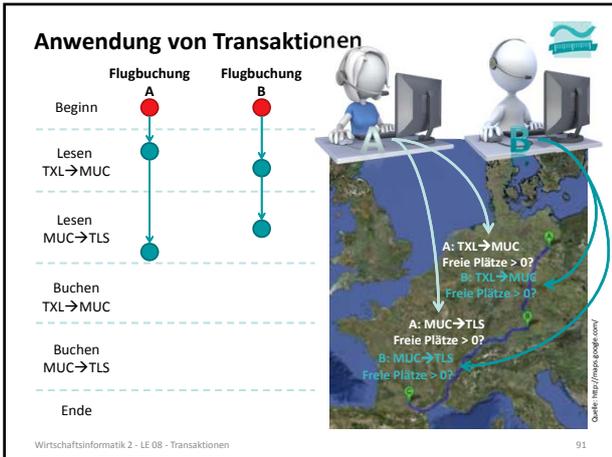
	Flugbuchung A	Flugbuchung B
Beginn	Red dot	Red dot
Lesen TXL → MUC	Teal dot	Teal dot
Lesen MUC → TLS	Teal dot	Teal dot
Buchen TXL → MUC	Teal dot	Teal dot
Buchen MUC → TLS	Teal dot	Teal dot
Ende	Red dot	Red dot

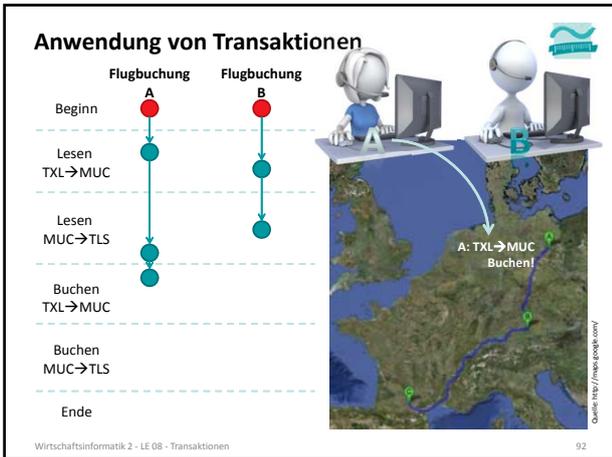
Wirtschaftsinformatik 2 - LE 08 - Transaktionen 87

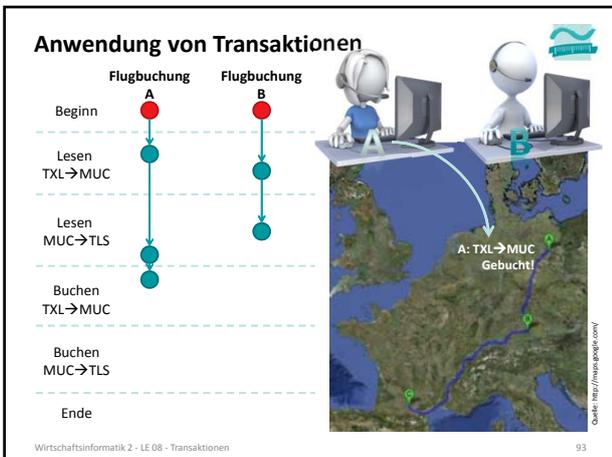


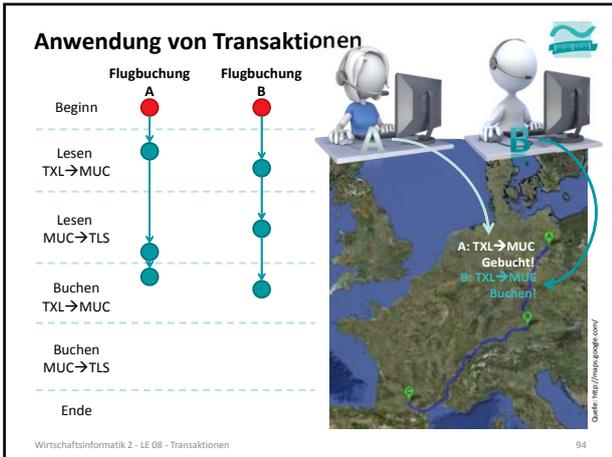


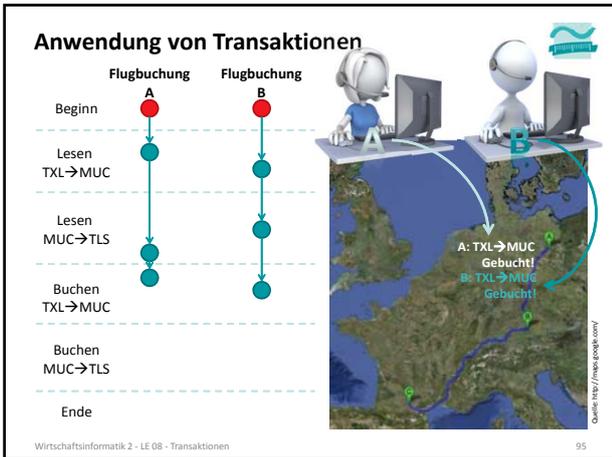


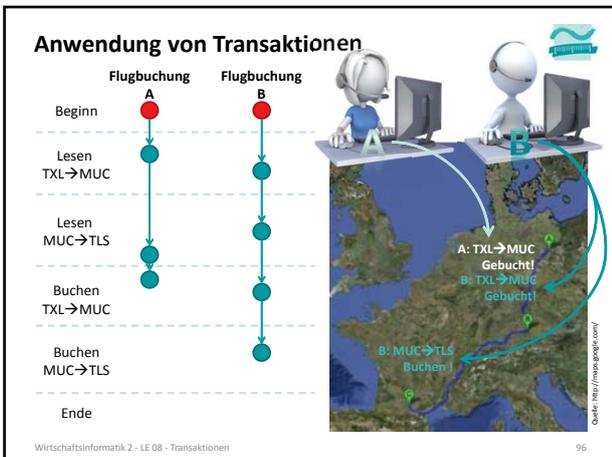


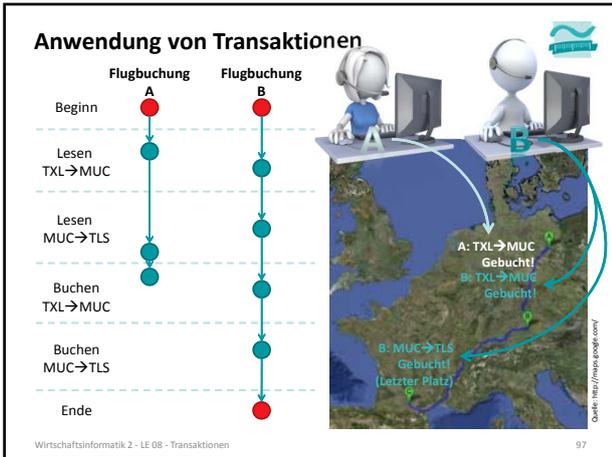


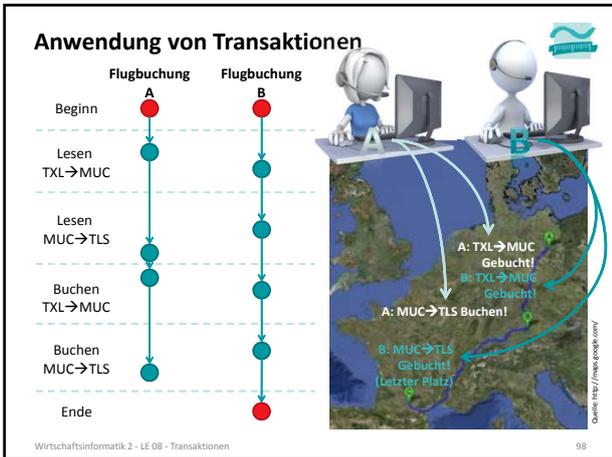


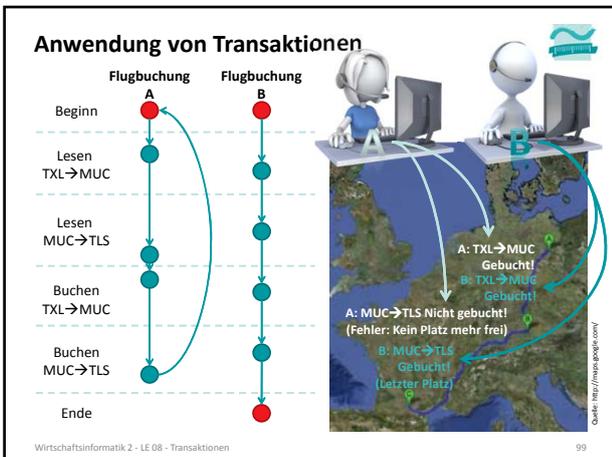


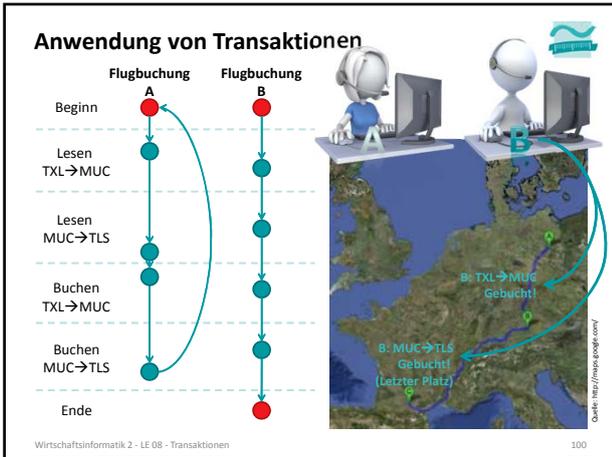


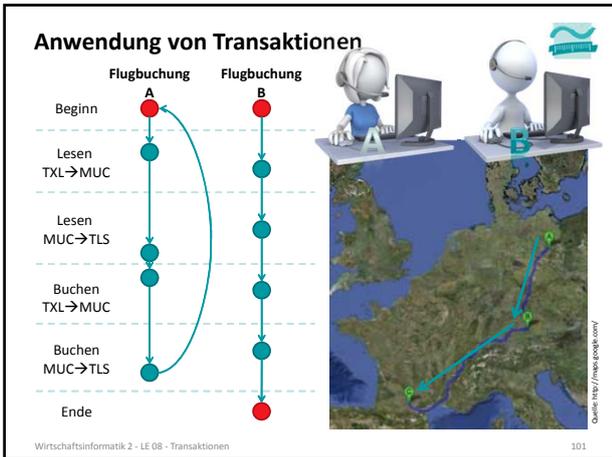












Transaktionen in SQL

SQL kennt besondere Befehle, mit denen Transaktionen

- begonnen
 - **BEGIN TRANSACTION** bzw. **START TRANSACTION**
 - Führt alle nachfolgenden SQL-Befehle innerhalb einer Transaktion aus
- erfolgreich abgeschlossen
 - **COMMIT** bzw. **COMMIT TRANSACTION**
 - schließt eine Transaktion ab, alle durchgeführten Operationen werden dauerhaft in der Datenbank wirksam
- nicht erfolgreich beendet
 - **ROLLBACK** bzw. **ROLLBACK TRANSACTION**
 - schließt die Transaktion ab, verwirft alle durchgeführten Operationen

werden können.

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 102

Transaktionen in SQL

Macht es aus Sicht des Anwenders Sinn ein einzelnes SELECT-, INSERT-, UPDATE- oder DELETE-Kommando in einer Transaktion auszuführen?

- einzelne Kommandos sind per Definition für sich genommen bereits atomar, konsistent, isoliert und dauerhaft
- sie müssen deshalb vom Programmierer nicht in einer Transaktion ausgeführt werden
- in einigen DBMS (z.B. Postgres) wird intern dennoch eine Transaktion auch für "atomare" Kommandos verwendet, weil diese auch viele Datensätze betreffen können (z.B. UPDATE)



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

103

Transaktionen in MS Access

Laut Dokumentation kennt MS Access die SQL-Anweisungen

- Beginn Transaction
- Commit Transaction
- Rollback Transaction

Es ist mir nicht gelungen, mit den Standard SQL-Befehlen in MS Access Transaktionen zu erzeugen.



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

104

Transaktionen in MS Access

Stattdessen

- Workspace-Objekt
 - Beginnen einer Transaktion
 - erfolgreichen Abschließen (Commit)
 - erfolglosem Beenden (Rollback)
- Datenbankfehler nicht verschlucken, sondern behandeln, insbesondere
 - Rollback
 - Fehlermeldung anzeigen

```

Sub demoTrans()
    On Error GoTo fehler
    'Deklaration
    Dim db As Database
    Dim wks As Workspace
    'Initialisierung
    Set db=CurrentDb
    Set wks=DBEngine.Workspaces(0)
    'Datenbankoperationen ausführen
    wks.BeginTrans
    db.Execute "<Irgendein SQL>"
    db.Execute "<Irgendein SQL>", _
        dbFailOnError
    '...
    wks.CommitTrans
    wks.Close
    Exit Sub

fehler:
    wks.Rollback
    wks.Close
End Sub
    
```

Wirtschaftsinformatik 2 - LE 08 - Transaktionen

105

Transaktionen in MS Access

Fehlerbehandlung

- bei Auftreten eines Fehlers Sprung zu einer Sprungmarke
- Hinter der Sprungmarke Befehle zur Behandlung des Fehlers ausführen, z.B.
 - Zurückrollen der Transaktion
 - Details zum Fehler dem Benutzer anzeigen, dazu kann das Err-Objekt verwendet werden

```

' Einschalten der Fehlerbehandlung durch Sprung
On Error GoTo fehler
' ...
fehler:
wks.Rollback ' Transaktion auf wks-Objekt
' Meldungsfenster mit Fehlerinformation
MsgBox Err.Description
' ...
            
```

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 106

Transaktionen in MS Access

Wichtig! Stolperfalle in Zusammenhang mit Fehlerverarbeitung:

- Symptom: On Error GoTo scheint ohne Wirkung zu bleiben, d.h. Fehler führen immer zur Unterbrechung des Programms, selbst wenn eine Fehlerbehandlung programmiert wurde
- Lösung: VBA-Editor > Menü "Extras" > Menüeintrag "Optionen" > Dialog "Optionen" > Registerkarte "Allgemein" > Gruppe "Unterbrechen bei Fehlern" > Option "Bei nicht verarbeiteten Fehlern" aktivieren



Wirtschaftsinformatik 2 - LE 08 - Transaktionen 107

Transaktionen in MS Access: Demo D08.01

D08.01: Tabelle und Formular

- Datenbank mit einer Tabelle "Konten", die Spalte für Saldo darf nicht negativ werden (Jugendkonto)
- In einem Formular
 - können ein Ausgangskonto und ein Zielkonto gewählt werden
 - ein Betrag kann erfasst werden und eine Überweisung getätigt werden




Wirtschaftsinformatik 2 - LE 08 - Transaktionen 108

Transaktionen in MS Access: Demo D08.01

D08.01: Modul "Konten"

- Gezeigt wird die Implementierung der Überweisung
 - zunächst ohne Transaktionen und dabei möglicherweise auftretende Probleme
 - dann mit Transaktionen und Vermeidung der Fehler
- Um den Effekt des Jugendkontos beobachten zu können, müssen wir erst Gutschreiben, dann Abbuchen!

```
Option Compare Database
Option Explicit

Private Sub ueberweisenOhneTransaktionen Me lngMktoKtoNo, lngMktoZtoNo, lngBetrag As Integer
    ' Prozedur ohne Transaktionen (Simulation)
    ' Prozedur mit Transaktionen (Simulation)
End Sub

Private Sub ueberweisenOhneTransaktionen Me lngMktoKtoNo, lngMktoZtoNo, lngBetrag As Integer
    ' Prozedur ohne Transaktionen (Simulation)
    ' Prozedur mit Transaktionen (Simulation)
End Sub
```

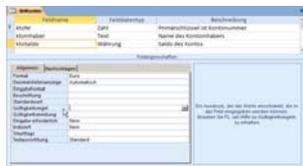
Wirtschaftsinformatik 2 - LE 08 - Transaktionen

109

Transaktionen in MS Access: Demo D08.01

D08.01: Tabelle "Kunden"

- Legen Sie eine Tabelle "tblKonten" an mit den Spalten
 - ktoNr: Typ Zahl (Long Integer), Primärschlüssel
 - ktoInhaber: Text
 - ktoSaldo: Währung
- Legen Sie als Gültigkeitsbedingung für die Spalte Währung fest, dass der Betrag nicht negativ sein darf
- Erfassen Sie Testdaten für die Tabelle



Wirtschaftsinformatik 2 - LE 08 - Transaktionen

110

Transaktionen in MS Access: Demo D08.01

D08.01: Umsetzung im Modul "Konten"

- Variante 1
 - Prozedur ueberweisenOhneTrans (siehe nächste Folie, im PDF-Format per Copy und Paste übernehmen) anlegen und analysieren
 - Einbinden der Prozedur in die Ereignisprozedur, die die Überweisung auslöst
 - führen Sie durch diesen Aufruf mehrere Transaktionen durch, die auch Fehlerfälle enthalten können
- Variante 2
 - Prozedur mit dem Namen ueberweisen implementieren, die Transaktionen nutzt und die Simulation eines Fehlers ermöglicht
 - Einbinden der Prozedur in die Ereignisprozedur, die die Überweisung auslöst
 - führen Sie durch diesen Aufruf mehrere Transaktionen durch, die auch Fehlerfälle enthalten können

Wirtschaftsinformatik 2 - LE 08 - Transaktionen

111

Inhalt

Ziel und Einordnung

Rückblick

Transaktionen

- Konsistenz und Integrität
 - Konsistenzsicherung als Ziel relationaler Datenbanken
 - Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung von Transaktionen
 - Anwendungsszenarien
 - Transaktionen in SQL
 - Transaktionen mit MS Access
- Technik der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 112

Parallele Transaktionen

Grundsätzlich gilt die ACID-Eigenschaft "Isolation"

- nach der sich parallele Transaktionen gegenseitig nicht beeinflussen dürfen
- DBS steht einem Benutzer nicht exklusiv zur Verfügung, deshalb
 - Antwortzeiten für jeden Benutzers sollen möglichst kurz sein
 - möglichst viele Benutzeraktionen sollen pro Zeiteinheit verarbeitet werden

Zur Leistungsoptimierung bringt DBMS die Einzeloperationen von parallelen Transaktionen in eine geeignete Reihenfolge (Schedule)

- Ergebnis paralleler Ausführung muss so sein, als wären Transaktionen nach einander ausgeführt worden
- auch neu eintreffende Transaktionen müssen mit bereits laufenden verzahnt werden
- nicht erfolgreich endende Transaktionen müssen beachtet werden

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 113

Parallele Transaktionen

Mangelnde Isolation kann zu folgenden Problemen führen¹

- Lost Updates: Zwei Transaktionen modifizieren parallel denselben Datensatz und nach Ablauf dieser beiden Transaktionen wird nur die Änderung von einer von ihnen übernommen
- Dirty Read: Daten einer noch nicht abgeschlossenen Transaktion werden von einer anderen Transaktion gelesen. Wird noch nicht abgeschlossene Transaktion anschließend zurückgesetzt, wurden von der anderen Transaktion falsche Daten gelesen.
- Non-Repeatable Read: Wiederholte Lesevorgänge liefern unterschiedliche Ergebnisse, weil zwischenzeitlich Änderungen committet wurden.
- Phantom Read: Suchkriterien treffen während einer Transaktion auf unterschiedliche Datensätze zu, weil eine (während des Ablaufs dieser Transaktion laufende) andere Transaktion Datensätze hinzugefügt oder entfernt hat.

¹ Quelle: http://de.wikipedia.org/wiki/Isolation_%28Datenbank%29
Wirtschaftsinformatik 2 - LE 08 - Transaktionen 114

Transaktionslevel

SQL-Standard definiert stufenartige Bedingungen (Level),

- mit denen Leistung und Isolation abgewogen werden können
- die Probleme mit Datenänderungen in parallelen Transaktionen unterschiedlich stark vermeiden helfen
 - Read Uncommitted
 - Read Committed
 - Repeatable Read
 - Serializable

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 115

Transaktionslevel

SQL-Standard definiert stufenartige Bedingungen (Level),

- mit denen Leistung und Isolation abgewogen werden können
- die Probleme mit Datenänderungen in parallelen Transaktionen unterschiedlich stark vermeiden helfen
 - Read Uncommitted
 - Read Committed
 - Repeatable Read
 - Serializable

Read Uncommitted

- jede Änderung wird sofort für alle anderen Transaktionen sichtbar (unabhängig davon, ob COMMIT zur Bestätigung bereits erfolgt ist)
- kann Dirty Read, Non-Repeatable Read und Phantom-Read verursachen
- kann bei sehr große Leseabfrage zur Reporterzeugung die Abfragegeschwindigkeit verbessern, wenn präzise Konsistenz nicht zwingend
- wird nicht von allen DBMS implementiert (z.B. nicht von MS Access, Postgres, aber von MySQL)

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 116

Transaktionslevel

SQL-Standard definiert stufenartige Bedingungen (Level),

- mit denen Leistung und Isolation abgewogen werden können
- die Probleme mit Datenänderungen in parallelen Transaktionen unterschiedlich stark vermeiden helfen
 - Read Uncommitted
 - Read Committed
 - Repeatable Read
 - Serializable

Read Committed

- lässt Änderungen für andere Transaktionen sichtbar werden, die vor der aktuell auszuführenden Operation committet wurden
- kann beim wiederholten Lesen der gleichen Daten ein anderes Ergebnis liefern (Non-Repeatable Reads oder Phantom Read)
- Standard-Level in MS Access

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 117

Transaktionslevel

SQL-Standard definiert stufenartige Bedingungen (Level),

- mit denen Leistung und Isolation abgewogen werden können
- die Probleme mit Datenänderungen in parallelen Transaktionen unterschiedlich stark vermeiden helfen
 - Read Uncommitted
 - Read Committed
 - **Repeatable Read**
 - Serializable

Repeatable Read

- wiederholte Leseoperationen auf gleichen Daten liefern immer die gleichen Ergebnisse
- hinzugefügte, gelöschte oder geänderte Datensätze bleiben unberücksichtigt (Phantom Read ist möglich)

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 118

Transaktionslevel

SQL-Standard definiert stufenartige Bedingungen (Level),

- mit denen Leistung und Isolation abgewogen werden können
- die Probleme mit Datenänderungen in parallelen Transaktionen unterschiedlich stark vermeiden helfen
 - Read Uncommitted
 - Read Committed
 - Repeatable Read
 - **Serializable**

Serializable

- parallel ablaufende Transaktionen liefern genau das gleiche Ergebnis, als würden sie nacheinander ablaufen
- obwohl es von außen so aussieht, dass Transaktionen nacheinander ablaufen, werden ihre Operationen intern tatsächlich parallel ausgeführt
- es können keine Probleme auftreten

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 119

Umsetzung paralleler Transaktionen

Verschiedene Mechanismen zur Umsetzung von Transaktionen

- Sperrverfahren: setzen während Schreib- und ggf. Leseoperationen eine Sperre auf Daten, so dass diese von parallelen Transaktionen nicht geändert werden können
 - einfache Sperrverfahren: sperren jeweils vor den Einzeloperationen und geben nach Einzeloperation die Sperre frei
 - 2-Phasen-Sperrverfahren: besorgen sich erst alle notwendigen Sperrern, führen dann alle Operationen durch und geben anschließend alle Sperrern wieder frei
- optimistische Verfahren: Transaktion wird durchgeführt und an ihrem Ende wird geprüft, ob es zu einem Konflikt gekommen sein könnte. In diesem Fall wird die Transaktion zurückgerollt andernfalls comittet
- Zeitmarkenverfahren: Zugriff auf Daten wird nur dann zugelassen, wenn eine bestimmte Verarbeitungsreihenfolge eingehalten wurde, andernfalls könnte ein Konflikt auftreten und deshalb wird die Transaktion zurückgesetzt

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 120

Protokollierung

Dauerhaftigkeit von Transaktionen erfordert, dass

- bei vollständiger oder teilweise Zerstörung der Datenbank der letzte konsistente Zustand wieder hergestellt werden kann
- bei Unterbrechung der Verarbeitung und damit u.U. laufender Transaktion (z.B durch Stromausfall) und anschließendem Neustart des Systems der letzte konsistente Zustand wieder hergestellt werden kann
- beim Zurücksetzen von Transaktionen der Zustand vor der Änderung wiederhergestellt werden kann

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 121

Protokollierung

Dauerhaftigkeit wird erreicht, indem Änderungen in einem Datenbank-Log protokolliert werden

Nr.	Schritt	Beschreibung
1	Lesen der Daten	Daten werden von der Festplatte gelesen und im Arbeitsspeicher bereitgestellt, sofern sie sich nicht bereits dort befinden
2	Merken bisheriger Daten	Zu ändernde Daten werden im Arbeitsspeicher zwischengespeichert (Before Image)
3	Ändern der Daten	Änderungen (Update, Insert, Delete) der Daten erfolgt im Arbeitsspeicher
4	Merken geänderter Daten	Geänderte Daten werden im Arbeitsspeicher zwischengespeichert (After Image)
5	Logdaten sichern	Before- und After Image werden vom Arbeitsspeicher in die Datei des Datenbank-Logs auf der Festplatte geschrieben
6	Geänderte Daten speichern	geänderten Daten werden vom Arbeitsspeicher in die Datenbank geschrieben
7	Transaktionsende	Commit-Eintrag wird in Log-Datei geschrieben. Anschließend wird der erfolgreiche Abschluss (dem Programm/Benutzer) bekannt gegeben

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 122

Protokollierung

Datenbank-Log dient zur (Re-)Konstruktion eines konsistenten Datenbankzustands

- wird eine Transaktion zurück gerollt, so kann anhand des Before-Image im Datenbank-Log der Ausgangszustand vor der Transaktion rekonstruiert werden
- wird die Datenbank vollständig oder teilweise vernichtet,
 - kann nach Einspielen der letzten Datenbanksicherung
 - das Datenbank-Log mit der Datenbank verglichen und
 - jede im Datenbank-Log abgeschlossene Transaktion in der Datenbank nachvollzogen werden (nicht abgeschlossene Transaktionen sind dann zurückgerollt)
 - dadurch entsteht der letzte konsistente Datenbankzustand
- darf nicht auf gleicher Festplatte gespeichert werden, wie Datenbank

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 123

Inhalt

Organisation

Rückblick

Einordnung

Transaktionen

- Konsistenzsicherung als Ziel relationaler Datenbanken
- Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung in SQL und mit MS Access
- Weitere Details der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 124

Zusammenfassung

Transaktionen

- sind zusammengehörige Abfolgen von Datenbankoperationen
- führen die Datenbank von einem konsistenten Zustand in einen konsistenten Zustand (nicht notwendigerweise ein anderer)
- weisen die ACID-Eigenschaften auf, d.h. sie stellen sicher
 - Atomarität
 - Konsistenz (Consistency)
 - Isoliertheit
 - Dauerhaftigkeit

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 125

Zusammenfassung

Transaktionen in MS Access

- Umsetzung basierend auf Workspace-Objekt des Benutzers mit

```
DBEngine.Workspaces(0).BeginTrans
DBEngine.Workspaces(0).CommitTrans
DBEngine.Workspaces(0).Rollback
```
- und der Option dbFailOnError beim Ausführen von, z.B.

```
CurrentDb.Execute "<einsQL>", dbFailOnError
```
- und der Option dbFailOnError beim Ausführen von, z.B.

```
Dim db AS Database
Set db = CurrentDb
Set rcs = db.OpenRecordSet("<einsQL>", dbOpenDynaSet, dbFailOnError)
```
- und einem Code-Block, in dem auf Fehler reagiert werden kann

```
On Error GoTo fehler
fehler:
```
- optional der Möglichkeit eigene Fehler (ab Fehlernummer 513) zu erzeugen und Fehler ausgeben

```
Err.Raise vbObjectError + 513, , "Eigene Fehlermeldung!"
MsgBox "Huch... " & Err.Description
```

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 126

Zusammenfassung

Technisch werden

- möglicherweise auftretende Probleme (z.B. Lost Update, Dirty Read, Unrepeatable Read und Phantom Read)
- durch Transaktionslevel (Read Uncommitted, Read Committed, Repeatable Read, Serializable) werden Probleme unterschiedlich konsequent ausgeschlossen

Umsetzung erfordert u.a.

- Synchronisationsverfahren (hier bspw. Sperrverfahren) und
- Datenbank-Log (hier insb. Before- und After-Image)

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 127

Inhalt

Ziel und Einordnung

Rückblicke

Transaktionen

- Konsistenzsicherung als Ziel relationaler Datenbanken
- Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung in SQL und mit MS Access
- Weitere Details der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 128

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 129



Inhalt

Ziel und Einordnung

Rückblicke

Transaktionen

- Konsistenzsicherung als Ziel relationaler Datenbanken
- Gefahren für die Konsistenz von Datenbanken
- Transaktionen und ACID
- Anwendung in SQL und mit MS Access
- Weitere Details der Transaktionsverarbeitung
 - Parallele Transaktionen und deren Probleme
 - Transaktionslevel
 - Umsetzung paralleler Transaktionen
 - Protokollierung
- Zusammenfassung

Ausblick

Wirtschaftsinformatik 2 - LE 08 - Transaktionen 130



BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

Wirtschaftsinformatik 2
LE 08 – Transaktionen

Prof. Dr. Thomas Off
<http://www.ThomasOff.de/lehre/beuth/wi2>
