Datenbankadministration

11. Synchronisation

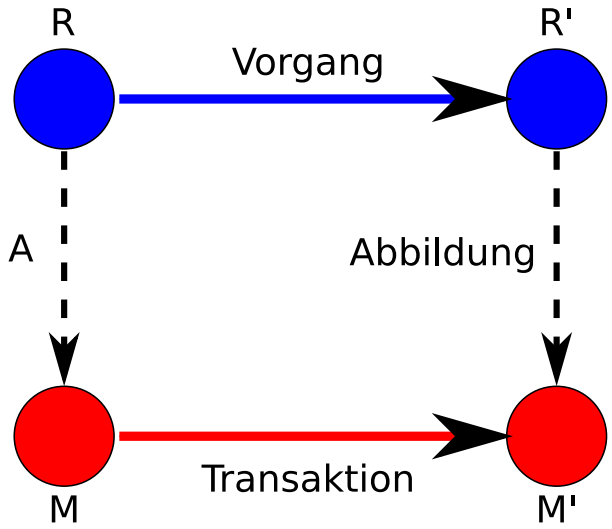
AG DBIS University of Kaiserslautern, Germany

Karsten Schmidt kschmidt@informatik.uni-kl.de
(Vorlage TU-Dresden)

Wintersemester 2008/2009

Transaktion

- **Transaktion**
  - Folge von Datenbankoperationen
  - bildet Vorgang in der modellierten Miniwelt ab
  - **alle** Zugriffe erfolgen durch Transaktionen



R: Realitätsausschnitt (Miniwelt)
M: Model der Miniwelt (beschrieben durch DB-Schema)
A: Abbildung aller wichtigen Objekte und Beziehungen
[Härder, Rahm 1999]

- **Probleme**
  - Nebenläufigkeit
  - Fehlersituationen (Systemfehler, Medienfehler, . . .)

ACID-Paradigma

- **Atomarität** (*atomicity*)
  - Alles-oder-Nichts-Prinzip
- **Konsistenz** (*consistency*)
  - logische Konsistenz
  - physische Konsistenz
  - gilt vor und nach der Transaktion
- **Isolation** (*isolation*)
  - virtueller Einbenutzerbetrieb
- **Dauerhaftigkeit** (*durability*)
  - Persistenz erfolgreicher Transaktionen

Anomalien & Isolationsstufen

Mehrbenutzeranomalien

- **Lost Update**
  - mehrere Transaktionen lesen & ändern dieselbe Ressource
  - nur letzte Änderung bleibt sichtbar
- **Dirty Read** (*uncommited read*)
  - Lesen von nicht festgeschriebenen Daten
  - nicht festgeschriebene Daten werden zurückgesetzt
- **Nonrepeatable Read**
  - mehrfaches Lesen derselben Ressource
  - unterschiedliche Werte bei wiederholtem Lesen
- **Phantom**
  - bei wiederholtem Lesen erfüllen mehr Tupel die Selektionsbedingung

Verhinderung von Mehrbenutzteranomalien

- **Isolationsstufen** (*isolation levels*)
  - Sperren von Daten für andere Transaktionen
  - Zielkonflikt: Nebenläufigkeit vs. Isolation
- **Setzen von Isolationsstufen**
  - vor Datenbankverbindung
    - `CHANGE ISOLATION TO {UR | CS | RS | RR}`
  - während Datenbankverbindung
    - `SET CURRENT ISOLATION {UR | CS | RS | RR | RESET}`
  - für einzelne Anfragen
    - `<sql-stmt> WITH {UR | CS | RS | RR}`

Isolationsstufen

- **Uncommited Read (UR)**
  - Sperren von Tupeln, falls andere Transaktion versucht, die zugrunde liegende Tabelle strukturell zu verändern oder zu löschen
  - Lesen von nicht festgeschriebenen Änderungen
  - Schreiben nur auf bereits festgeschriebene Änderungen
  - Anomalien
    - Dirty Read, Nonrepeatable Read, Phantom
  - für Transaktionen
    - auf Tabellen, auf denen nicht geschrieben wird
    - für die Lesen unbestätigter Änderungen unkritisch ist

Isolationsstufen

- **Cursor Stability (CS)**
  - Sperren des vom Cursor **aktuell** referenzierten Tupels
  - Freigabe bei
    - Repositionierung des Cursors (mittels `FETCH`)
    - Schließen des Cursors bzw. Transaktionsende
  - zusätzliches Sperren **aller** veränderter Tupel
  - Anomalien
    - Nonrepeatable Read, Phantom
  - Standard-Isolationsstufe
- **Beispiel**
  - ```
    DECLARE C1 CURSOR FOR
    SELECT R_NAME, R_COMMENT FROM REGION;

    OPEN C1;
    FETCH C1 INTO name, comment;
    CLOSE C1;
    ```
    ```

Isolationsstufen

- **Read Stability (RS)**
  - Sperren aller **gelesenen** oder **geänderten** Tupel für gesamte Transaktion
  - Anomalien
    - Phantom
- **Repeatable Read (RR)**
  - Sperren aller **referenzierten** Tupel für gesamte Transaktion
  - Anomalien: keine

Sperrverwaltung

Sperren

- **Sperren**
  - Zuordnung von Ressourcen mit Transaktion
  - Ziel: Zugriff anderer Transaktion beschränken
    - Sperrenvergabe bei Zugriff auf Ressource
    - inkompatibler Zugriffsversuch $\rightarrow$ Warten bis sperrende Transaktion beendet
  - Attribute
    - `OBJECT`: gesperrte Ressource
    - `DURATION`: Zeitraum der Sperrenaktivität (vgl. Isolationsstufe)
    - `MODE`: Art des Zugriffs für Sperrinhaber und konkurrierende Transaktionen

Sperrmodi

- **Share (S)**
  - gültiger Objekttyp: Tabelle, Tupel

	Lesen	Schreiben
Sperrinhaber	✓	✘
konkurrierende TA	✓	✘

- **Intent Share (IS)**
  - gültiger Objekttyp: Tabellenbereich, Tabelle

	Lesen	Schreiben
Sperrinhaber	✓	✘
konkurrierende TA	✓	✓

  - Sperrinhaber: S-Sperre auf gelesene Tupel
- **Beispiel**
  - `SET CURRENT ISOLATION RS`
  - `SELECT * FROM REGION WHERE R_REGIONKEY=1`

Sperrmodi

- **Exclusive (X)**
  - gültiger Objekttyp: Tabelle, Tupel, Pufferpool

	Lesen	Schreiben
Sperrinhaber	✓	✓
konkurrierende TA	(✓)	✘

  - konkurrierende TA: lesender Zugriff nur mit UR-Isolationsstufe
- **Intent Exclusive (IX)**
  - gültiger Objekttyp: Tabellenbereich, Tabelle

	Lesen	Schreiben
Sperrinhaber	✓	✓
konkurrierende TA	✓	✓

  - Sperrinhaber: S-Sperre auf gelesene Tupel, X- und U-Sperre auf geschriebene Tupel
- **Beispiel**
  - `SET CURRENT ISOLATION CS`
  - `UPDATE REGION SET R_NAME='NEW_NAME' WHERE R_REGIONKEY=1`

Sperrmodi

- **Share with Intent Exclusive (SIX)**
  - gültiger Objekttyp: Tabelle

	Lesen	Schreiben
Sperrinhaber	✓	✓
konkurrierende TA	✓	✘

  - Sperrinhaber: X-Sperre auf geschriebene Tupel, **keine** Sperre auf gelesene Tupel
- **Update (U)**
  - gültiger Objekttyp: Tabelle, Tupel

	Lesen	Schreiben
Sperrinhaber	✓	✓
konkurrierende TA	✓	✘

  - Sperrinhaber: X-Sperre auf geschriebene Tupel
- **Beispiel**
  - `SET CURRENT ISOLATION RS`
  - `SELECT * FROM REGION FOR UPDATE`

Sperrmodi

- **Super Exclusive (Z)**
  - gültiger Objekttyp: Tabellenbereich, Tabelle
  - Sperrinhaber: lesender und schreibender Zugriff, Tabelle ändern oder löschen, Index für Tabelle erstellen oder löschen, Tabellenreorganisation
  - andere: kein Zugriff
  - Beispiel
    - `ALTER TABLE REGION ADD COLUMN NEW_COLUMN INT`
- **Weitere Sperren**
  - Intent None (IN)
  - Next Key Share (NS)
  - Next Key Exclusive (NX)
  - Next Key Weak Exclusive (NW)
  - Weak Exclusive (W)

Sperrkompatibilitätsmatrix

Compatibility of Page and Row Lock Modes			
Held Lock	Requested Lock		
	S	U	X
S	Yes	Yes	No
U	Yes	No	No
X	No	No	No

Compatibility of Table and Table Space Lock Modes						
Held Lock	Requested Lock					
	IS	IX	S	U	SIX	X
IS	Yes	Yes	Yes	Yes	Yes	No
IX	Yes	Yes	No	No	No	No
S	Yes	No	Yes	Yes	No	No
U	Yes	No	Yes	No	No	No
SIX	Yes	No	No	No	No	No
X	No	No	No	No	No	No

Sperrverwaltung

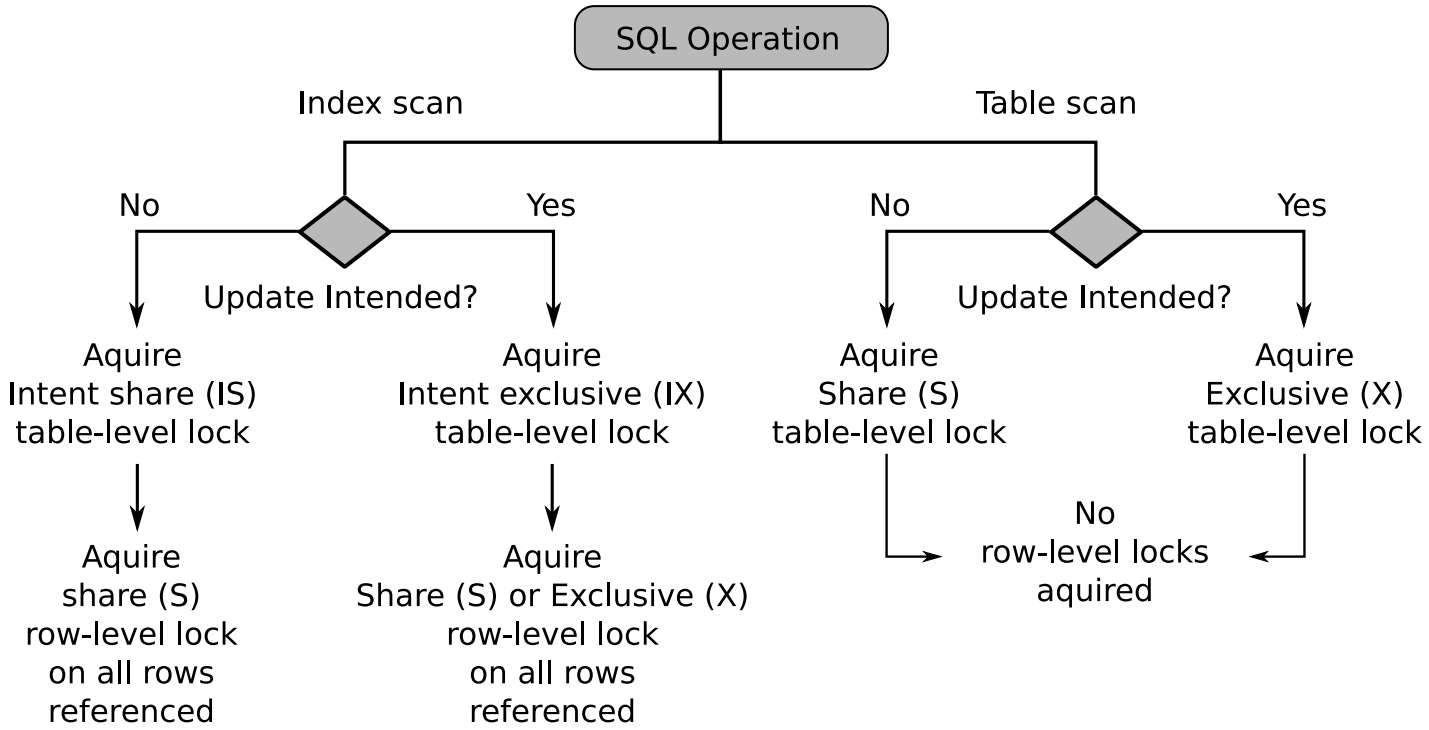
- **Sperrumwandlung** (*lock conversion*)
  - maximal eine Sperre pro Transaktion und Ressource
  - restriktivere Sperren ersetzen aktuelle Sperren
  - Ausnahme: S + IX $\rightarrow$ SIX
- **Sperreneskalation** (*lock escalation*)
  - Problem: begrenzter Speicherbereich
  - Anzahl der Sperren durch Datenbankkonfigurationsparameter `LOCKLIST` und `MAXLOCKS` beschränkt
  - Überschreitung
    - Sperren auf Tupelebene $\rightarrow$ Sperren auf Tabellenebene
    - Fehler, falls nicht ausreichend

Sperrverwaltung

- **Verklemmung** (*deadlocks*)
  - Situation: 2 Transaktionen warten auf gegenseitige Sperrfreigabe
  - Erkennung durch das Datenbanksystem (*deadlock detector*)
  - Datenbankkonfigurationsparameter `DLCHKTIME`
  - Zurücksetzen einer der beiden Transaktionen
- **Sperrzeiten** (*lock timeout*)
  - Verhinderung von langen Wartezeiten
  - Lösung
    - Datenbankkonfigurationsparameter `LOCKTIMEOUT`
    - Rücksetzen der anfordernden Transaktion

Sperrverwaltung

- **Sperranforderung**
  - implizite Vergabe und Verwaltung
  - Sperren auf Tupelebene durch DB2 bevorzugt
  - Erzwingen von Sperren auf Tabellenebene
    - `ALTER TABLE <table-name> LOCKSIZE TABLE`
  - Explizite Sperranforderung
    - `LOCK TABLE <table-name> IN [SHARE|EXCLUSIVE] MODE`



Zusammenfassung

- **Transaktionen**
  - ACID-Paradigma
- **Mehrbenutzerbetrieb**
  - Lost Update
  - Dirty Read
  - Nonrepeatable Read
  - Phantom
- **Synchronisation**
  - Isolationslevel
  - Sperren