Im Multiuserbetrieb können strukturell bedingte Fehler auftreten, die durch gleichzeitigen und/oder voneinander abhängigen Datenzugriff entstehen. Eine Aufgabe des DBMS besteht darin. Schutzmechanismen für diese Fälle bereit zu stellen.

Fehlerart 1 (gleichzeitiger Datenzugriff)

User A und User B lesen den selben Datenbestand, um das Ergebnis in weiteren SQL-States für Manipulationsabfragen zu nutzen.

Zeit	User A	User B
1	# Wert Lesen (Ergebnis: `@tmp = 10`) `SELECT @tmp := col1 FROM tab1 ;`	
2		# Wert Lesen (Ergebnis: `@tmp = 10`) `SELECT @tmp := col1 FROM tab1 ;`
3	# Wert verarbeiten (Ziel: `col1= 30`) `UPDATE tab1 SET col1 = col1 + 2 * @tmp ;`	
4		# Wert verarbeiten (Ziel: `col1= 15`) `UPDATE tab1 SET col1 = col1 + 0.5 * @tmp ;`
5		# Kontrolle (Ergebnis: `@tmp = 35`) `SELECT col1 FROM tab1 ;`

(User A möchte das Doppelte , User B die Hälfte zu einem vorhandenen Wert addieren)

Durch den (quasi) gleichzeitigen Zugriff auf denselben Datenbestand erzielen weder User A noch User B das gewünschte Resultat ihrer Datenmanipulation!

Fehlerart 2 (Zusammengehörige SQL-States)

User A führt mehrere Befehle aus die, nur vollständig, wiederum zu einem konsistentem Datenbankzustand führen (Überweisen von 20 Euro von einem auf das andere Konto).

Zeit	User A
1	# Überweisung Speichern `INSERT INTO tab2 (1,2,20.00 €);`
2	# Geld auf Zielkonto gutschreiben `UPDATE tab1 SET col1 = col1 + 20 WHERE Konto = 1;`
3	# Geld auf dem Lastkonto abbuchen `UPDATE tab1 SET col1 = col1 – 20 WHERE Konto = 2;`

Sollten nicht alle Befehle ausgeführt werden, (z.B. Verbindungsabbruch oder auf dem Konto 1 keine Deckung vorliegt) befindet sich die DB in einem inkonsistenten Zustand!

In MySQL sind zwei verschiedene Schutzmechanismen (Locking→Fehler1 und Transaktion→Fehler2) notwendig, die nur teilweise miteinander kombiniert werden können. Transaktionen besitzen vier Eigenschaften, die man auch ACID-Kompatibilität nennt.

A	**tomicity** (untrennbar)	Eine TA stellt eine **untrennbare** Einheit von Anweisungen dar, die nicht einzeln ausgeführt werden (“Alles oder Nichts !”).
C	**onsistency** (widerspruchsfrei)	Die Datenbasis befindet sich vor und nach einer TA im **widerspruchsfreien** Zustand.
I	**solation** (vereinzelt)	Mehrere Operationen oder TA’s werden **vereinzelt** abgearbeitet und beeinflussen sich nicht gegenseitig.
D	**urability** (dauerhaft)	Wenn eine TA fehlerfrei ausgeführt wurde, werden die Daten **dauerhaft** in der Datenbasis gespeichert

Locking realisiert das zeitweilige Sperren von Tabellen oder Datensätzen für den exklusiven Zugriff durch einzelne User (oder Prozesse). Alle Manipulationen werden sofort permanent gespeichert!

automatische Transaktionen Transaktionen werden momentan nur von den InnoDB und BerkleyDB – Tabellen unterstützt! Mit der optionalen Angabe des Isolationsgrades kann die TA gegen andere Zugriffe abgegrenzt werden. Per Default (Standardwert = REPEATABLE READ) sehen alle anderen User die Datenbasis im Zustand vor Transaktionsbeginn (oder SERIALIZABLE gar nicht). Weiterhin muss der `AUTOCOMMIT`-Modus abgeschaltet werden, der jedes einzelne SQL-State als TA behandelt.

TA Syntax Teil 1

```
①  [SET [GLOBAL | SESSION] TRANSACTION ISOLATION LEVEL
②                  { READ UNCOMMITTED | READ COMMITED |
③                    REPEATABLE READ | SERIALIZABLE } ]
④  [SET AUTOCOMMIT = 0;]
⑤  START TRANSACTION;
⑥        Anweisung(en)
⑦        :
⑧  COMMIT;
⑨  [SET AUTOCOMMIT = 1;]
```

Hinweis

Bei dieser Variante wird der Ablauf komplett vom DBMS überwacht. Bei Fehlern (z.B. Verbindungsabbruch o.Ä.) wird automatisch ein ROLLBACK ausgeführt und alle Anweisungen widerrufen. Die Datenbasis befindet sich danach im

gesteuerte Transaktionen Der Verlauf einer Transaktion kann mit dem Befehlen **COMMIT** oder **ROLLBACK** gesteuert, dass heißt in Abhängigkeit von Ereignissen ausgeführt oder widerrufen werden. Um eine bedingte Verzweigung zu realisieren, wird der Befehl **IF...THEN...END IF** verwendet, der nur in Stored Proceduren eingesetzt werden kann.

TA Syntax Teil 2

```
①  . . . Deklaration der Stored Procedure
②  START TRANSACTION;
③        Anweisung(en)
④        :
⑤        IF “Bedingung = i.O.“ THEN
⑥              COMMIT;
⑦        ELSE
⑧              ROLLBACK;
⑨        END IF
⑩  . . . Abschluss der Stored Procedure
```

Hinweise

Bei dieser Variante wird der Ablauf (ausführen oder widerrufen der Transaktion) durch das Ergebnis der bedingten Verzweigung gesteuert. Verbindungsabbruch bewirkt ein ROLLBACK.

Tabellen Sperren Mit Locking wird auch für MyISAM Tabellen unterstützt.

Locking

```
①  LOCK TABLES Tabelle1 { READ|READ LOCAL|WRITE } [,Tabelle2 Locktype … ];
②        Anweisung(en)
③        :
④  UNLOCK TABLES;
```

Hinweise

Grundsätzlich müssen alle verwendeten Tabellen (auch nur gelesene) gesperrt werden.

READ	Alle dürfen lesen, aber keiner (auch nicht “LOCK-Ausführer”) darf Daten verändern.
READ LOCAL	Wie READ , allerdings sind INSERT’s erlaubt.
WRITE	User darf lesen und schreiben, alle anderen sind vollständig blockiert.

Informationen über den aktuellen Zustand erhält man aus den Systemvariablen:

```
SELECT @@autocommit , @@tx_isolation , @@global.tx_isolatin ;
```