**Transaktionen**

**Ein Konzept, das gewährleistet, dass zusammenhängende Operationen entweder GANZ oder GAR NICHT ausgeführt werden.**

**Stichwort: Datenkonsistenz und Datensicherheit**

**Beispiel**

Prozess des Abbuchens **von Girokonto auf Sparkonto**

Betrag 1 EUR wird von Girokonto weggenommen

Betrag 1 EUR wird Sparkonto gutgeschrieben

**Ablauf OK.**

Kontostand Girokonto: um 1 EUR weniger

Kontostand Sparkonto: um 1 EUR mehr.

**Beispiel**

Betrag 1 EUR wird von Girokonto weggenommen

FEHLER / ABRRUCH / etc.

Betrag 1 EUR wird dem Sparkonto NICHT gutgeschrieben

**Ablauf NICHT OK.**

Kontostand Girokonto: um 1 EUR weniger

Kontostand Sparkonto: gleich

**Fazit: 1 EUR ist verschwunden.**

**Beispiel**

Betrag 1 EUR wird von Girokonto weggenommen

FEHLER / ABRRUCH / etc.

Betrag 1 EUR wird Sparkonto NICHT gutgeschrieben

TRANSAKTION macht ROLLBACK, Vorgang des Abbuchens vom Girokonto wird nicht COMMITet

Ablauf NICHT OK **ABER**

Datenkonsistenz durch Transaktion beibehalten und vor allem kein Geldverlust.

* **Implizite Transaktionen erklären**
* **Explizite (programmierte) Transaktionen erklären**

**Genereller Aufbau**

-- SET XACT\_ABORT ON - sorgt für automatisches Rollback

BEGIN TRANSACTION

-- Anweisung 1

-- Anweisung 2

-- ......

-- Fehler?

-- Automatisches ROLLBACK

-- Sonst:

COMMIT TRANSACTION

**SET XACT\_ABORT ON** -- sorgt für automatisches Rollback

**BEGIN TRANSACTION**

UPDATE Kunde

SET Email = 'eva@eee.de'

WHERE KundeId = 2007

UPDATE Kunde

SET Email = 'aaa'

WHERE KundeId = 3

**COMMIT TRANSACTION**

**BEGIN TRANSACTION**

UPDATE Kunde

SET Email = 'eva@eee.de'

WHERE KundeId = 2007

*-- Vor Rollback (Wert innerhalb der Transaktion bereits geändert, aber noch nicht commitet*

SELECT Email FROM Kunde WHERE KundeId = 2007

**ROLLBACK TRANSACTION** -- (hier nur als Beispiel)

-- Transaction wurde durch Rollback verworfen (nicht commitet). **Alter Wert bleibt erhalten**

SELECT Email FROM Kunde WHERE KundeId = 2007