FH JOANNEUM
University of Applied Sciences

DIHSÜD

Relationale Datenbanken

Muesluem Atas

Internet-Technologien & - Anwendungen

02.03.2022

Über mich

Müslüm Atas

Internet-Technologien & - Anwendungen
FH JOANNEUM GmbH
Werk-VI-Strasse 46, Raum 014
8605 Kapfenberg

muesluem.atas@fh-joanneum.at

- Bachelor- & Masterstudium - Telematik:
  - @TU Graz
  - IT-Sicherheit und Software Entwicklung
- Doktorat der technischen Wissenschaften
  - Group Decision Technologies for Complex Products and Services
- Forschungsschwerpunkte und Lehre
  - Relational Database & Database Design
  - Mobile Application Development
  - Recommender- & Configuration Systems

Agenda

- Einführung in relationale Datenbanksysteme
- Structured Query Language
- Datenschutz
- Transaktionen
- SQL Injections

Motivation

- Stark zunehmende Informationsflut
- Die Daten müssen verwaltet werden!
- Kein Mensch ist in der Lage, die Daten in vernünftiger Zeit mit vernünftigem Aufwand zu strukturieren
- Weitere Probleme:
  - Redundanz und Inkonsistenz
  - Beschränkte Zugriffsmöglichkeiten
  - Verlust von Daten
  - Sicherheitsprobleme
  - Probleme beim Mehrbenutzerbetrieb

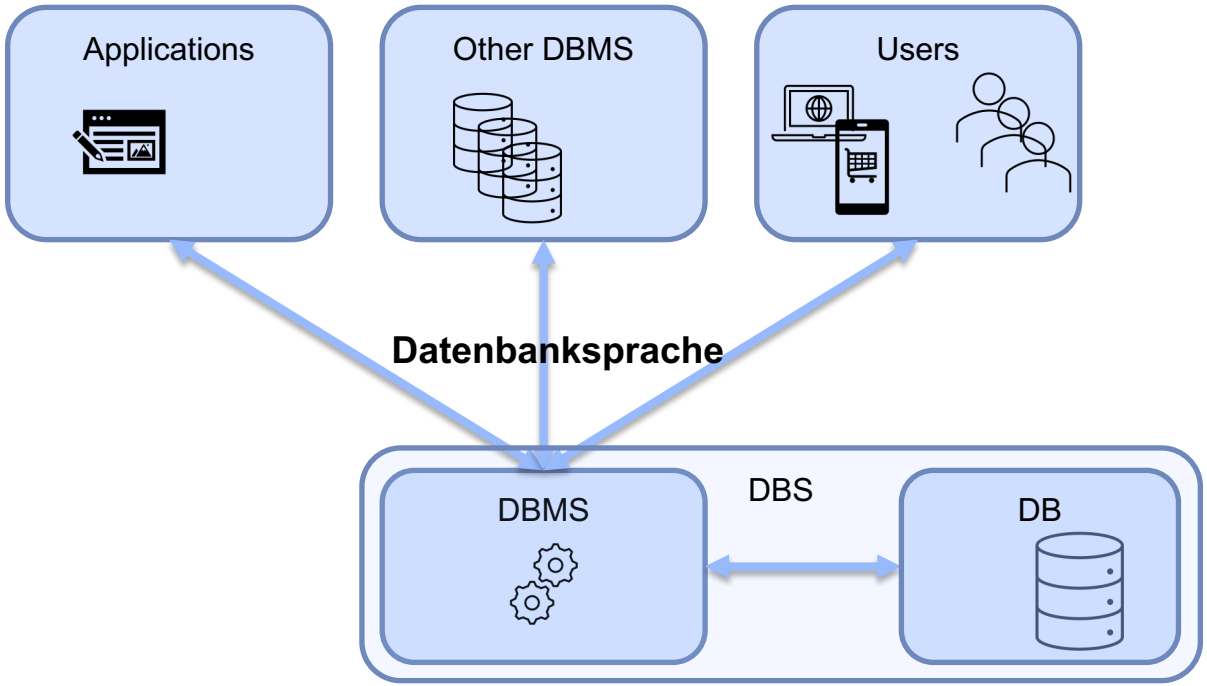
DB Definition

„*Eine Datenbank ist eine selbständige und auf Dauer ausgelegte Datenorganisation, welche einen Daten bestand sicher und flexibel verwalten kann.*“

Rene Steiner, 2014

- Eine Datenbank (DB) soll:
  - Beliebige Daten verwalten (Einfügen, Löschen, Nachführen)
  - Informationen aus diesen Daten liefern
  - Unberechtigten Personen den Zugriff auf die Daten verweigern

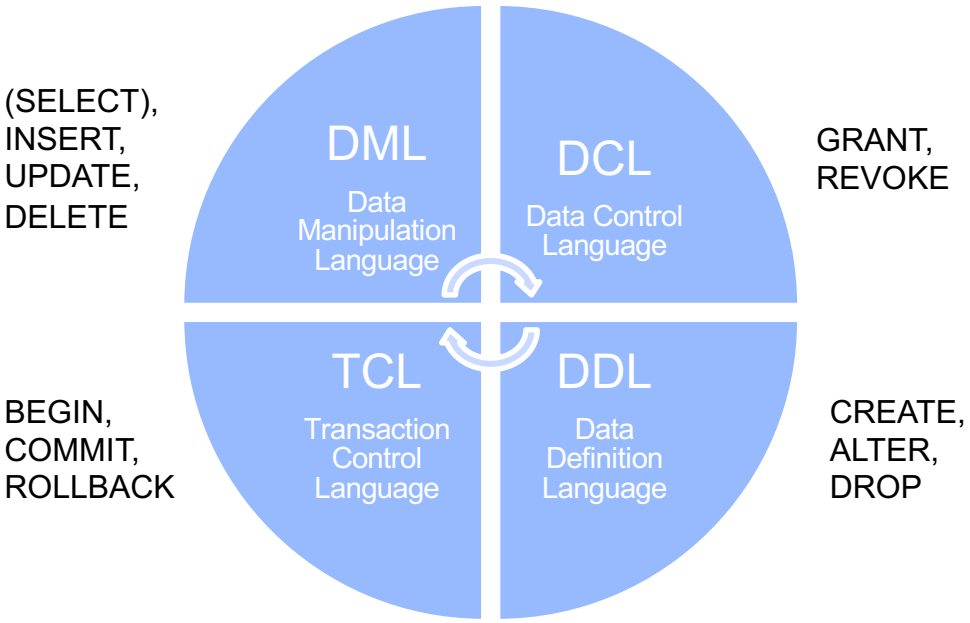
Datenbank Bestandteile



Datenbanksprache

SQL (Structured Query Language)

- Schnittstelle zwischen Benutzer/Applikation und DBMS



Datenbankverwaltungssystem

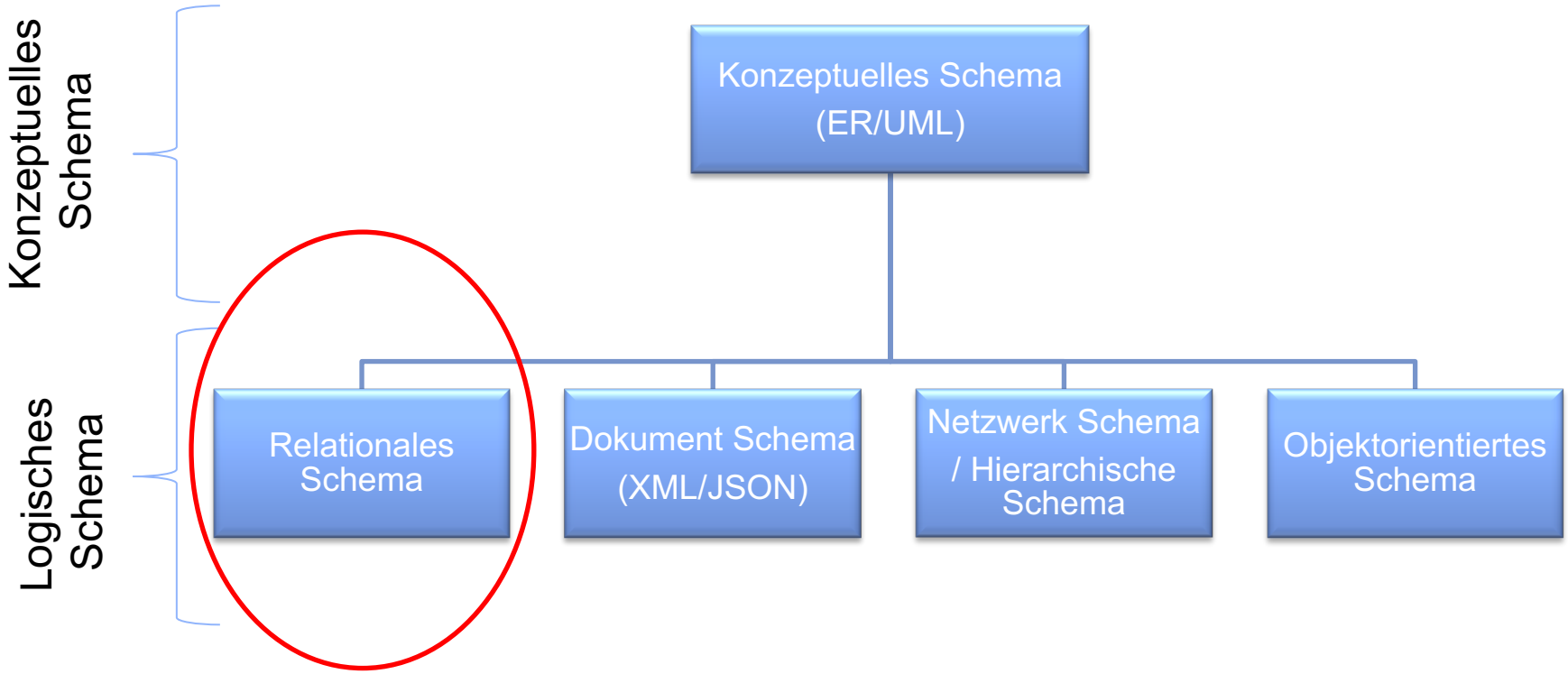
Database Management System (DBMS)

- Kern der Datenbank

- Beinhaltet alle notwendigen Systemroutinen für Datenbankfunktionen:
  - Suchen
  - Lesen
  - Schreiben
  - Löschen
  - Updaten

- Programme können nur über definierte Schnittstellen des DBMS auf die gespeicherten Daten zugreifen

Datenbank-Modelle



Relationale Datenbanken

- **R**elationale **D**aten**b**anken (**RDB**)
- Die Daten werden nach Themenkreisen (Entitäten) in Form von Tabellen angelegt
- Beispiel: **Kursverwaltung**

Personaldaten

PNr	Name
121	Meier
134	Steffen
155	Huber

Kursdaten

KNr	Titel
100	abc
102	def
105	xyz

Kursbesuche

PNr	KNr	Datum
121	100	1.1.2005
121	102	10.3.2005
121	102	1.2.2006
134	100	3.4.2007
155	105	4.4.2008
155	105	1.8.2008

Wichtigste Begriffe der RDB

Personen ← Entität (Tabellenname)

Attribute →

Tupel (Datensatz) →

PNr.	Name	Vorname	Größe	Geschlecht
1234	Müller	Hans	182	m
5634	Suter	Ernst		m
2456	Tarelli	Claudia	170	w
1123	Brunner	Diana	172	w

↑ Attributwerte

Beziehungen zwischen den Tabellen (I)

- Beziehungen zwischen den Tabellen durch:
  - Primärschlüssel *(engl. primary key-PK)*
  - Fremdschlüssel *(engl. foreign key-FK)*

- **Primärschlüssel** Kriterien:
  - Jedes Tupel muss *unique* sein
  - Jedem neuen Tupel muss sofort ein Attributwert zugeteilt werden können
  - Der Schlüsselwert darf sich während dessen Existenz nicht ändern
  - Es kann mehr als einen Schlüssel für einen Entity-Typ geben
  - Kann auch künstlich erzeugt werden

(natürlicher) Primärschlüssel

Personen

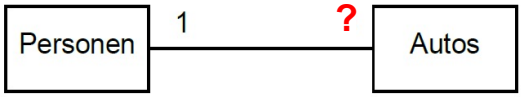
PersNr	Vorname	Name	…
234	Müslüm	Atas	…
235	Hans	Winter	…
236	Thomas	Sauer	…

(künstlicher) Primärschlüssel

Autos

ANr	Marke	*PersNr*
1	Audi A3	234
2	VW Golf	236

Beziehungen zwischen den Tabellen (II)



Abkürzung	Assoziationstyp	Anzahl Tupel der Tabelle 2
1	einfache Assoziation	genau ein Tupel (1)
c	konditionelle Assoziation	kein **oder** genau ein Tupel (0/1)
m	multiple Assoziation	mindestens ein Tupel (≥1)
mc	multipel-konditionelle Assoziation	beliebig viele Tupel (≥0)

Typ 1: Jede Person besitzt genau ein Auto

Typ c: Eine Person kann ein oder kein Auto besitzen

Typ m: Jede Person besitzt mindestens ein Auto

Typ mc: Eine Person kann beliebig viele Autos besitzen (auch keines)

Fremdschlüssel

- Ein Attribut, das auf einen Datensatz in einer anderen Tabelle verweist

(natürlicher) Primärschlüssel

Personen

PersNr	Vorname	Name	…
234	Müslüm	Atas	…
235	Hans	Winter	…
236	Thomas	Sauer	…

Fremdschlüssel

(künstlicher) Primärschlüssel

Autos

ANr	Marke	*PersNr*
1	Audi A3	234
2	VW Golf	236

Structured Query Language (SQL)

SQL-Anweisungen

- SQL ist case-**in**sensitive!
- SQL wird von den meisten DBSs unterstützt
  - ORACLE
  - MySQL
  - IBM DB2
  - Microsoft ACCESS
  - SQL-Server
- Dokumentation:
  - https://dev.mysql.com/doc/
- Beispiel:

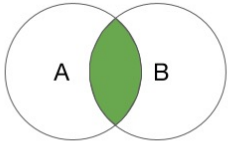
```
CREATE TABLE personen (
  vorname VARCHAR(20) NOT NULL,
  …,
  svnr NUMERIC(10) NOT NULL);
```

SQL-Element	SQL-Anweisung	Beschreibung
Datendefinition	CREATE TABLE	Tabelle erzeugen
	ALTER TABLE	Tabelle ändern, Beziehungen verwalten
	DROP TABLE	Tabelle löschen
	CREATE INDEX	Index erstellen
	DROP INDEX	Index löschen
Datenmanipulation	INSERT INTO	Datensätze einfügen
	UPDATE	Datensätze nachführen
	DELETE	Datensätze löschen
Datenabfrage	SELECT	Datensätze abfragen
Datenschutz	CREATE ROLE	Rollen erzeugen
	DROP ROLE	Rollen löschen
	SET ROLE	Rollen aktivieren
	GRANT	Berechtigungen vergeben
	REVOKE	Berechtigungen entziehen
Transaktionen	COMMIT	Änderungen verbuchen
	ROLLBACK	Änderungen verwerfen

Weitere SQL Befehle

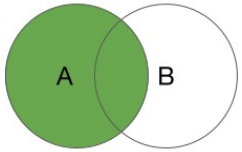
- Tabellen verknüpfen mit **JOIN**:

```
SELECT * FROM t1 JOIN t2 ON t1.join_attribute = t2.join_attribute;

SELECT * FROM t1 INNER JOIN t2 ON t1.join_attribute = t2.join_attribute;

SELECT * FROM t1, t2 WHERE t1.join_attribute = t2.join_attribute;
```

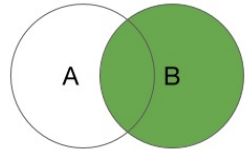


- **LEFT/RIGHT OUTER JOIN** :

```
SELECT t1.*, t2.* FROM t1 LEFT OUTER JOIN t2
ON t1.join_attribute = t2.join_attribute;
```



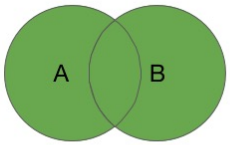
LEFT OUTER JOIN



RIGHT OUTER JOIN

- **FULL OUTER JOIN** bzw. **UNION**:

```
SELECT * FROM t1 FULL OUTER JOIN t2 ON t1.join_attribute = t2.join_attribute;

(SELECT * FROM t1 LEFT OUTER JOIN t2 ON t1.join_attribute = t2.join_attribute)
UNION
(SELECT * FROM t1 RIGHT OUTER JOIN t2 ON t1.join_attribute = t2.join_attribute);
```

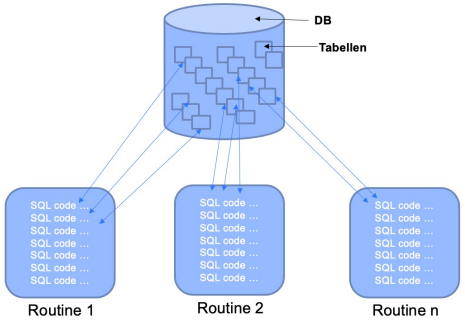


http://kirillpavlov.com/blog/2016/04/23/beyond-traditional-join-with-apache-spark/, 01.11.2018

Datenschutz in SQL

- Berechtigungsverwaltung von Benutzer/Rollen
- Views
- Stored Routines
  - Stored Procedures
  - Stored Functions
- Trigger

Berechtigungs-gruppen	Benutzer 1 (Personaldienst)				Benutzer 2 (Verkäufer)			
Tabellen	**A**	**E**	**B**	**L**	**A**	**E**	**B**	**L**
Personen	X	X	X	X	X			
Autos	X				X	X	X	X



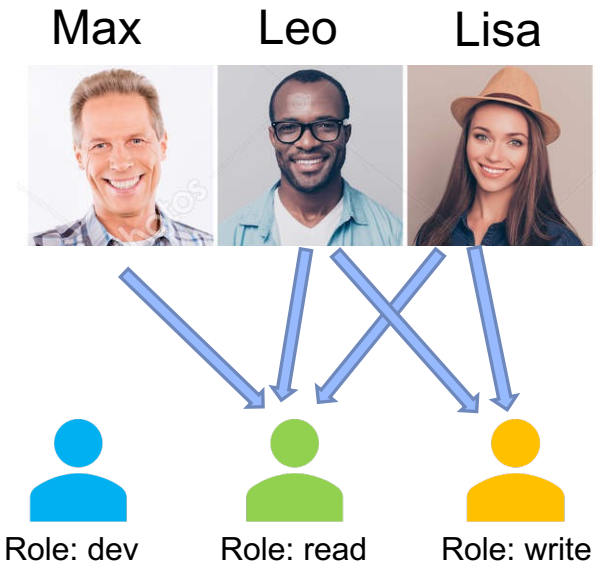
Verwaltung von Berechtigungen

- Benutzer erstellen und Rollen zuweisen:

```
CREATE USER max IDENTIFIED BY {secure_pw1};
CREATE USER leo IDENTIFIED BY {secure_pw2};
CREATE USER lisa IDENTIFIED BY {secure_pw3};
```

- Rollen definieren & Berechtigungen vergeben:

```
CREATE ROLE dev, read, write;

GRANT ALL ON db.* TO dev;
GRANT SELECT ON db.table(attr1, attr2) TO read;
GRANT INSERT, UPDATE, DELETE ON db.* TO write;


GRANT read TO max;
GRANT read, write TO leo, lisa;
```



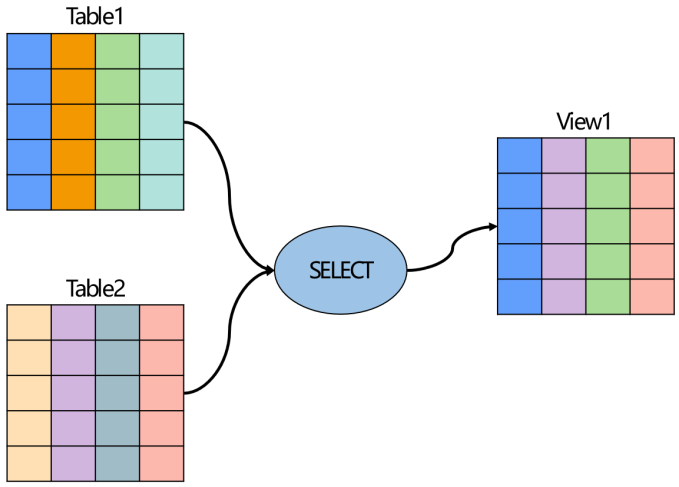
https://st3.depositphotos.com/4678277/19081/i/1600/depositphotos_190818136-stock-photo-collage-diverse-people-expressing-positive.jpg

Views

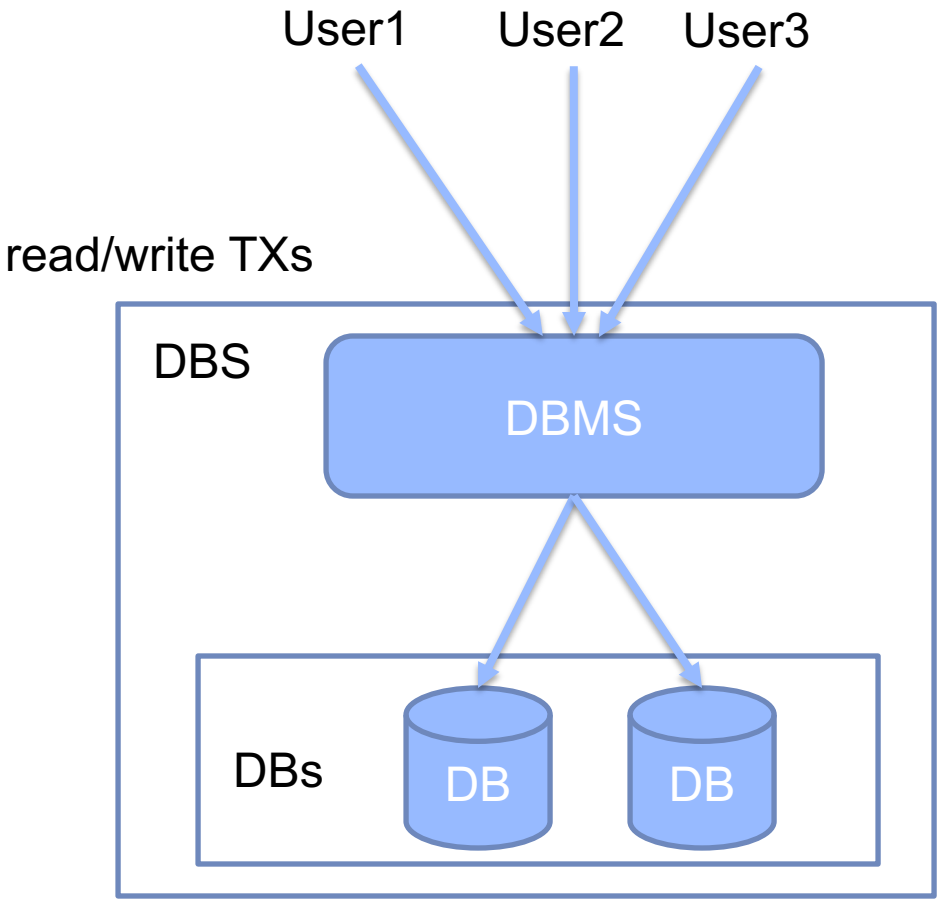
View: Eine Pseudotabelle

- Beispiel:

```
CREATE VIEW v_persons AS SELECT name, vorname, PLZ, Ort FROM personen;

GRANT SELECT ON v_persons TO Sekretariate;
GRANT INSERT ON v_persons TO Personaldienst;

INSERT INTO v_persons (Name, Vorname, PLZ, Ort)
VALUES ('Müller', 'Hans‘, 1050‘, ‚Wien‘);
```



Transaktionen (TXs)



1. **Mehrere Benutzer**

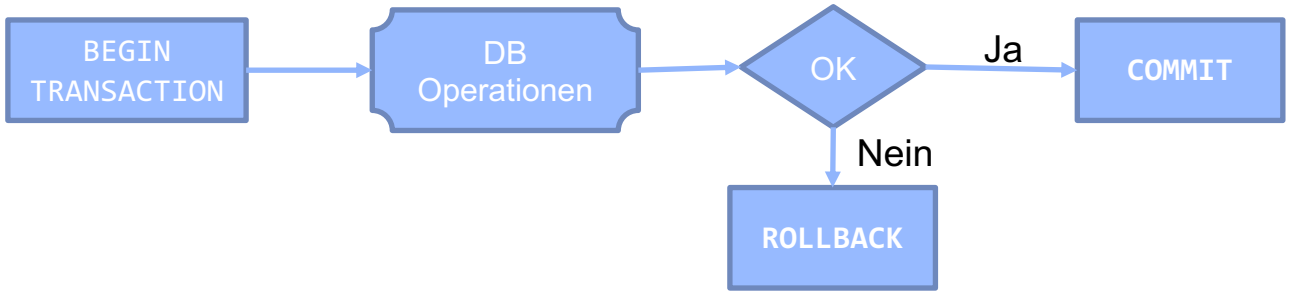
2. **Diverse Fehler**
- Deadlocks
- Netzwerk Fehler
- Stromausfall
- Etc.

Transaktionen (I)

- Mechanismen, die verhindern sollen, dass die Daten der DB inkonsistent werden
- Idee: Mehrere Operationen am Datenbestand zu einer Transaktion zusammenzufassen
- Anforderungen:
  - Wiederherstellung des vormals korrekten Datenzustands (*Recovery*) mit Hilfe der Transaktionen
  - **ACID-Prinzipien** erfüllen
    - **A**tomicity: Alles oder nichts!
    - **C**onsistency: DB in einem konsistenten Zustand hinterlassen
    - **I**solation: Nebenläufige Transaktionen beeinflussen sich (logisch) nicht
    - **D**urability: Die Ergebnisse persistent speichern

Transaktionen (II)

- Anforderungen an den Programmierer einer Transaktion:
  - Transaktionen sollten möglichst kurze Laufzeiten haben
  - Datensätze sollen nur gesperrt werden, wenn dies wirklich notwendig ist



Beispiel:

- Absturz nach Schritt 3?
  - Folgen:
    Konto A wird reduziert, ohne dass Konto B jemals erhöht wurde

Nr.	Transfer von 50€ von Konto A nach Konto B	
1.	Lese den Kontostand von A in die Variable a	`read(A,a);`
2.	Reduziere den Kontostand um 50 €	`a:= a-50;`
3.	Schreibe den neuen Kontostand	`write (A,a);`
4.	Lese den Kontostand von B in die Variable b	`read(B,b);`
5.	Erhöhe den Kontostand um 50 €	`b:= b + 50;`
6.	Schreibe den neuen Kontostand	`write(B, b);`

SQL-Injection

dt. *SQL-Einschleusung*

- Das Ausnutzen einer Sicherheitslücke in Datenbankensystemen
- Sonderfunktionen für den SQL-Interpreter
  - Erlauben die externe Beeinflussung der ausgeführten Befehle
    - Doppelter Bindestrich, Anführungszeichen, Semikolon, …
- Oft tritt in Zusammenhang mit PHP- und ASP-Programmen auf
- Weltweit beliebteste Methode, da einfach und schnell ausgeführt
- Grundregel: „*Traue nie den Eingaben des Benutzers*“

Ich habe im Browser am Ende der URL ein paar Zeichen eingefügt und begehe jetzt ein Kapitalverbrechen, weil ich unerlaubt auf ein Computersystem zugreife.“

Aktuelle Beispiele

- SQL-Angriffe machen 27% aller Web-Angriffe aus
- *LinkedIn* (Juni 2021) - 6,5 Millionen Kennwörter gestohlen
  - Ein Angriff führte zu unmittelbaren Wiederherstellungskosten von 1 M$ + 2-3 M$ für notwendige Upgrades
- *Yahoo* (Juli 2012) - Mehr als 450.000 Anmeldedaten
  - https://www.zdnet.com/article/450000-user-passwords-leaked-in-yahoo-breach/
- *British Royal Navy* (2010) – Die Webseite wurde geschädigt
- *Sony* (Juni 2008)- Persönlichen Daten von Millionen von Nutzern, Gutscheinen, Download-Schlüsseln und Kennwörtern

https://www.akamai.com 01.02.2019
www.zdnet.com

SQL-Injection Beispiele

- Skriptzeilen (Pseudocode) für den Webserver-Zugriff

```
1  uname = request.POST['username']
2  passwd = request.POST['password']
3
4  sql = "SELECT id FROM users WHERE username='" + uname + "' AND password='" + passwd + "'"
5
6  database.execute(sql)
```

- Der Angreifer hat die Möglichkeit, per SQL-Injection das **Passwortfeld zu manipulieren**

```
1  sql = "SELECT id FROM users WHERE username='' AND password='password' OR 1='1'
```

- Eingabefeld:

- Resultat:

```
SELECT * FROM Users WHERE UserId = 105; DROP TABLE Suppliers;
```

SQL-Injection verhindern

Schritt 1: Automatische Eingaben der Applikationen überwachen

- Übergebene Daten sollten immer in erwarteten Datentyp vorliegen
  - Beispiel: Numerischer Parameter mit `is_numeric()`-Funktion überprüfen

Schritt 2: Umfassender Server-Schutz

- Nur notwendige Anwendungen/Diensten installieren bzw. aktivieren
  - Löschen von allen nicht gebrauchten Benutzerkonten

Schritt 3: Datenbank härten und sichere Codes verwenden

- DB von irrelevanten Faktoren befreien und regelmäßig aktualisieren
- Speichern aller sensiblen Daten in verschlüsselter Form
- Hash-Funktionen: `md5(), sha1(), PASSWORD(),ENCRYPT()`

https://www.ionos.at, 01.02.2019

Literatur - Referenz

- **Date, C. J.:** *An Introduction to Database Systems. Addison Wesley*
- **Kemper, A., Eickler, A.:** *Datenbanksysteme. Eine Einführung. De Gruyter Oldenbourg*
- **Stickel, E.:** *Konzeptuelle Datenmodellierung. Teubner*
- **Schubert, M.:** *Datenbanken. Theorie, Entwurf und Programmierung relationaler Datenbanken. Vieweg+Teubner*
- **Silberschatz, A., Korth, H. F., Sudarshan, S.:** *Database System Concepts McGraw Hill*
- **Steiner, R.:** *Grundkurs Relationale Datenbanken. Vieweg*
- **Stickel, E.:** *Datenbankdesign: Methoden und Übungen. Gabler*
- **Zehnder, A.:** *Informationssysteme und Datenbanken. vdf-Hochschulverlag*
- **Mata-Toledo, R. A., Cushman, P. K.:** *Schaum's outline of Fundamentals of Relational Databases. McGraw-Hil*
- *Datenbanksysteme: Eine Einführung Unterlagen von Technische Universität München,* https://db.in.tum.de/teaching/bookDBMSeinf/folien/index.shtml?lang=de, https://db.in.tum.de/teaching/ws1516/grundlagen/, 01.09.2018
- *Datenbanksysteme 1 Unterlagen von Universität Potsdam,* https://hpi.de/fileadmin/user_upload/fachgebiete/naumann/folien/SS11/DBS_I/DBS1_03_RelationalerEntwurf.pdf, 01.09.2018
- *Slideshare, Einstieg in relationale Datenbanken mit MySQL (Folien),* https://www.slideshare.net/titanoboa/einstieg-in-relationale-datenbanken-mit-mysql-folien, 01.09.2018
- *Relationale Datenbanken (ITM, SS16) Unterlagen von FH-Joanneum,* 17.09.2018
- *Datenbanken (OM7/WI7 1. Sem.) Unterlagen von Hochschule der Medien (HdM) Stuttgart*, https://www.hdm-stuttgart.de/~riekert/lehre/, 20.09.2018

Feedback

https://fragebogen.joanneum.at/dihsued_feedback/?q=base&r=AV283221