Datenbanken

HS Bhven, FB1, Studiengang Medizintechnik
WS 23/24, MT-B 7

richard rascher-friesenhausen
richard.rascher-friesenhausen@hs-bremerhaven.de

20. Oktober 2023

Inhalt

Organisatorisches
Einleitung
Daten, Datenbanken, Datenbankmanagementsysteme
Datenbankentwurf
Entity-Relationship-Modell
Relationales Datenbankschema, Tabellen
Anomalien und Normalformen
SQL
Relationen-Algebra
SQL — Teil 2

Organisatorisches

- ‚*Termin*‘:
  Vorlesung: Mi, 09:45 – 11:15, Raum C303/304
  Übung: Mi, 11:30 – 13:00, MSM

- ‚*Kontakt*‘: am besten über Email
  `richard.rascher-friesenhausen@hs-bremerhaven.de`
  Sprechstunde in Z1230 oder Online nach Vereinbarung.

- ‚*Leistungsnachweis*‘: über regelmäßige Abgabe von Hausaufgaben. In Gruppen zu maximal 2 Teilnehmern. Zettel siehe Webseite zur Veranstaltung.

Für die Vorlesung ist kein Buch erforderlich. Soll alles an die Tafel, auf Folie oder wird ausgeteilt. Aber wer denn doch eins in der Hand halten will:

1. Günter Matthiessen: ‚*Relationale Datenbanken und Standard-SQL*‘, Addison-Wesley, 2008
2. Alfons Kemper: ‚*Datenbanksysteme. Eine Einführung.*‘, Oldenburg, 2009
3. Jay A. Kreibich: ‚*Using SQLite*‘, O'Reilly, 2010
4. . . . und vieles im Netz.

Einleitung

Thema der Vorlesung ‚*Datenbanken*'. Also eine Bank für Daten.

Was sind Daten? Was ist Information? Was ist Wissen?

‚*Daten*': sind repräsentierte Fakten.

‚*Informationen*': sind interpretierte Daten.

‚*Wissen*': ist verknüpfte Informationen.

Bsp.:

- Datum: 36.6.
- Information: Körpertemperatur.
- Wissen: Normaltemperatur

Wir bleiben alleine bei den Daten.

Beispiele für ‚Datenbanken':

- Telefonbuch
- Literaturverzeichnis
- Amazon.de
- PACS
- www.physionet.org
- www.medline.de
- . . .

Einleitung

Erwartungen an eine Datenbank: (etwa Amazon.de, google.de)

- Daten sollen nicht-flüchtig (persistent) sein.
  Speicherung auf Festplatte, optischen Medien, ... Nicht im Hauptspeicher. Beim Ausschalten eines Rechners muss darauf laufende Datenbank erhalten bleiben.
- Es sollen beliebig viele Daten abgelegt werden können.
  Größe einer Datenbank häufig nicht im Vorhinein bekannt. Und es kann immer noch etwas dazu kommen...
- Auf Daten soll sehr schnell zugegriffen werden können (lesen, einfügen, ändern, löschen).
  Bei Anfragen möchte man nicht lange warten.
- Daten sollen stets verfügbar sein, etwa 24/7.
  Auch ein Frage der Server.
- Daten sollen korrekt und vollständig sein.
  Beachten von verbindlichen Forderungen und Prüfungen.

Einleitung

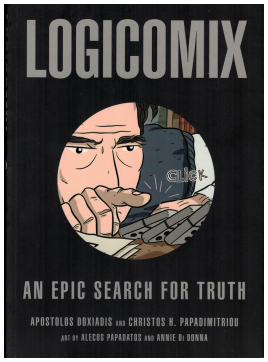
Erwartungen an eine Datenbank (Fortsetzung): (etwa Amazon.de, google.de)

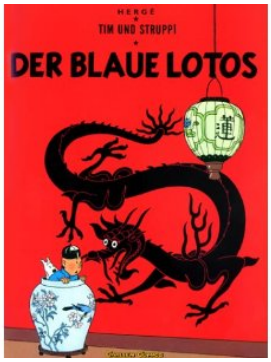
- Daten sollen einfach zugreifbar sein.
  Man muss kein Informatiker sein, um die Datenbank nutzen zu können.
- Daten sollen flexibel zugreifbar sein.
  Auch komplexe Anfragen sollen beantwortet werden können.
- Daten sollen sicher sein gegen Verlust.
  Nicht einfach überschreibbar, Backups, ...
- Daten sollen sicher sein gegen unberechtigten Zugriff.
  FireWall, Benutzerkontrolle, ...
- Daten sollen von vielen gleichzeitig abrufbar sein.
  Parallelbetrieb.
- ...

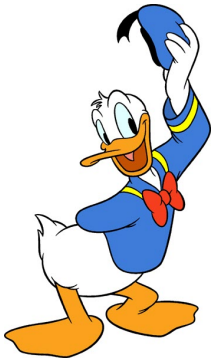
Nicht nur Forderungen an eine Datenbank als Software, sondern auch an Hardware.

Konkretes Beispiel:

Comic-Sammlung & Ausleihe an Freunde





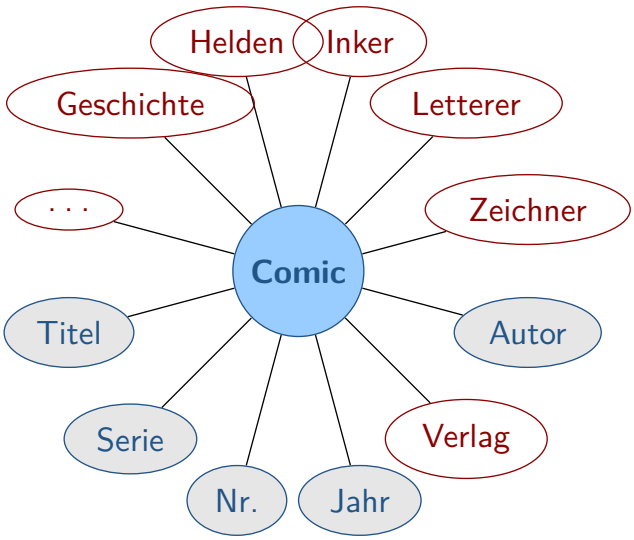


Was würde man sich zu einem ‚*Comic*' merken wollen?
Was würde man sich zu einer ‚*Freundin*' merken wollen?
Was würde man sich zu einer ‚*Ausleihe*' merken wollen?

Konkretes Beispiel:

Comic-Sammlung & Ausleihe an Freunde

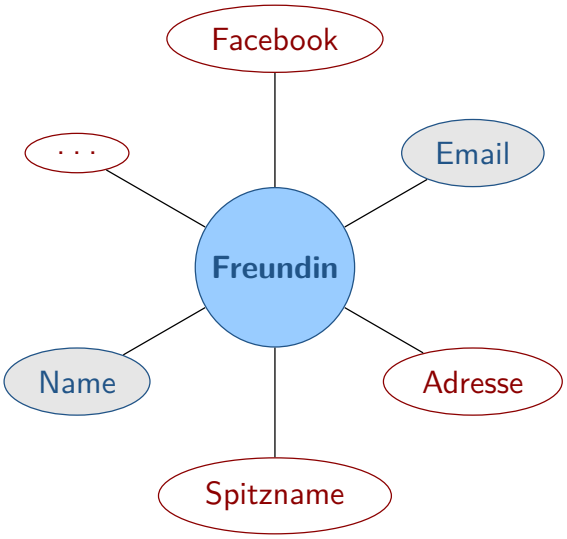
Was würde man sich zu einem ‚*Comic*' merken wollen?



Konkretes Beispiel:

Comic-Sammlung & Ausleihe an Freunde

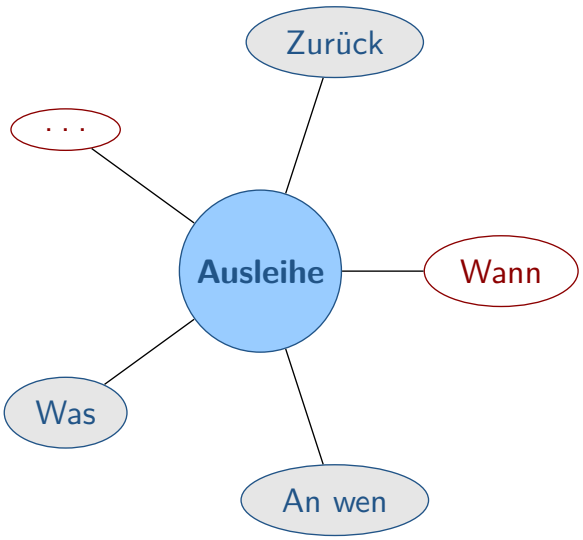
Was würde man sich zu einer *‚Freundin'* merken wollen?



Konkretes Beispiel:

Comic-Sammlung & Ausleihe an Freunde

Was würde man sich zu einer *‚Ausleihe'* merken wollen?



Erste Idee dazu: analog
‚*Karteikasten*' mit ‚*Karteikarten*' auf dem Schreibtisch.
Zweite Idee dazu: digital
‚*Dateisystem*' und ‚*Dateien*' auf dem Computer.

‚*Datei*': Daten Kartei

Benötigen mindestens drei Dateien: `comics.txt`, `friends.txt`, `rentals.txt`.

Datei `comics.txt`:

```
! Tim & Struppi * Der blaue Lotos * 4 * 1934 * Herge
! Tim & Struppi * Tim im Kongo * 1 * 1930 * Herge
! Sleeper * Out in the Cold * 1 * 2003 * Ed Brubaker
! Sleeper * All False Moves * 2 * 2004 * Ed Brubaker
! - * Logicomix: An Epic Search for Truth * - * 2010 * A. Doxiadis, C. Papadimitriou
```

Spezielles selbst ausgedachtes ASCII Datenformat für Serie, Titel, Nr., Jahr, Autor.
! leitet Datensatz ein. * trennt Datenfelder innerhalb Datensatz.

Datei `friends.txt`:

```
! Tim * tim@gmx.de
! Struppi * struppi@google.com
! Apostolos * apostolos.doxiadis@whereever.gr
```

Tabelle mit Vorname, EMail.

Datei `rentals.txt`:

```
! Struppi * Logicomix * Nein
! Ed * Sleeper-1 * Ja
! Apostolos * Tim&Struppi-2 * Ja
```

Tabelle mit An wen, Was, Zurück.

Zu den Textdateien mit den Daten können noch Programme und Formulare hinzu kommen, die die Verwaltung erleichtern:
Eingeben, Aktualisieren, Lesen, Löschen, Suchen, Statistik, ...
für jede einzelne Datei. Heute meist Web-basiert.

Einleitung

Dateibasierte Datenhaltung für kleine Datenmengen oder einfach strukturierte Daten möglich und sinnvoll. Aber . . .

Probleme mit dateibasierter Datenbank:

- Datenformat.
- Datenintegrität (Korrektheit und Vollständigkeit).
- Redundanz (Überfluss).
- . . .

Einleitung

Inhalte der Vorlesung:

1. Definition von Daten, Datenbanken und Datenbankmanagementsystemen
2. Datenbankentwurf
3. Normalformen
4. SQL
5. SQLite

Daten, Datenbanken, Datenbankmanagementsysteme

Daten
Datenbanken
Datenbankmanagementsysteme

Daten

Wikipedia sagt

> *‚Daten‘*: sind zum Zweck der Verarbeitung zusammengefasste Zeichen, die aufgrund bekannter oder unterstellter Abmachungen Informationen tragen.

Bspe.:

- `21.10.2021` als Datum von heute,
- `TK` als Abkürzung von ‚Techniker Krankenkasse‘,
- ...

Repräsentation von Fakten, Konzepten oder Instruktionen in formalisierter Art und Weise.
Ideal für Computer (und Datenbanken).
Man unterscheidet zum einen

strukturierte, semistrukturierte und unstrukturierte Daten.

Sowie

transiente und persistente Daten.

Über Datentypen reden wir, wenn wir uns SQL anschauen.

Strukturierte Daten

Daten, die in ein festgelegtes Formblatt hineinpassen. Reihenfolge, Typ und Länge der Daten vorgeschrieben. Einfach maschinell verarbeitbar.

```
!␣Tim␣&␣Struppi␣*␣Der␣blaue␣Lotos␣*␣4␣*␣1934␣*␣Herge
!␣Tim␣&␣Struppi␣*␣Tim␣im␣Kongo␣␣␣␣*␣1␣*␣1930␣*␣Herge
!␣Sleeper␣␣␣␣␣␣␣*␣Out␣in␣the␣Cold␣*␣1␣*␣2003␣*␣Ed␣Brubaker
!␣Sleeper␣␣␣␣␣␣␣*␣All␣False␣Moves␣*␣2␣*␣2004␣*␣Ed␣Brubaker
```

Vorgegebene Längen lassen sich auch durch Trennzeichen ersetzen (hier ! und *). Oder CSV (comma separated values).

Form, Inhalte, Bedeutung müssen gesondert von Daten abgelegt werden und vor der Verarbeitung der Daten bekannt sein.

Semistrukturierte Daten

Form ist nicht im vor hinein festgeschrieben, wird mit den Daten abgelegt. Beispiele sind XML (eXtendible Markup Language) oder YAML (Yet Another Markup Language) Dateien.
Form ist flexibel, lässt sich bei Einhaltung der Vorgaben aber ebenfalls einfach maschinell verarbeiten.

```
<comic>
␣␣<serie>Tim␣&␣Struppi</serie>
␣␣<titel>Der␣blaue␣Lotos</titel>␣<nr>4</nr>
␣␣<jahr>1934</jahr>
␣␣<autor>Herge</autor>
</comic>

<comic>
␣␣<titel>Out␣in␣the␣Cold</titel>␣<nr>1</nr>
␣␣<serie>Sleeper</serie>␣<autor>Ed␣Brubaker</autor>
␣␣<jahr>2003</jahr>
</comic>
```

Unstrukturierte Daten

Keine Vorgaben der Form der Daten. Beispiele sind gescannte Handschriften, Tonaufnahmen natürlicher Sprache, ... Verarbeitung jeweils speziell.
Machen wir nicht.

Soweit zur Form der Daten. Jetzt zur Lebenszeit der Daten.

Transiente Daten

Die Daten werden kurzfristig von einem Prozess angelegt. Nach Prozessende werden die Daten gelöscht und sind nicht mehr zugreifbar. Dies gilt bspw. für berechnete Daten eines Computerprogramms, die im flüchtigen Hauptspeicher abgelegt werden. Nach dem Ende des Programms wird der Hauptspeicher neu zwischen den laufenden Programmen aufgeteilt.

Persistente Daten

Die Daten sind dauerhaft abgelegt und können von verschiedenen Prozessen bearbeitet werden. Beispielsweise kann das Computerprogramm berechnete Daten in eine Datei auf die Festplatte schreiben. Die Werte sind dann bei einem Neustart wieder zugänglich oder können von einem anderen Programm geplottet werden.

Datenbanken stellen sicher, dass die Daten persistent sind.

Datenbanken

‚Datenbank‘: (DB) persistente, strukturierte Sammlung von Datensätzen einschließlich notwendiger Metainformationen: Indices, Kataloge, Integritätsbedingungen.

- ▷ *‚Index‘* beschreibt Zugriffspfad auf die Daten,
- ▷ *‚Katalog‘* enthält Informationen über Feldlängen, Feldtypen, Zugriffsrechte, . . . ,
- ▷ *‚Integritätsbedingungen‘* beschreiben Anforderungen an die Daten.

Bsp.: Zettelkasten mit Englisch-Vokabeln. Kartenvorderseite: deutsches Wort, Kartenrückseite: englisches Wort.

- Indizierung über alphabetische Sortierung,
- Katalog sagt, jeweils deutsches und englisches Wort,
- Integrität bedeutet, zu jedem deutschen Wort muss mindestens ein englisches gehören (und umgekehrt).

Einzelnes Datum ohne Bedeutung. Durch die Angabe, wofür es steht, wird ein Zusammenhang zur Anwendung klar.
Daten in Tabellen sammeln. Spaltenüberschriften und Tabellentitel (*‚Datenbankschema‘*) stellen Anwendungszusammenhang her.

Comics

Serie	Titel	Nr	Jahr	Autor
Tim&Struppi	Der blaue Lotos	4	1934	Herge
Tim&Struppi	Tim im Kongo	1	1930	Herge
-	Logicomix	-	2010	Doxiadis
Sleeper	Out in the Cold	1	2003	Ed Brubaker
Sleeper	All False Moves	2	2004	Ed Brubaker

Einzelne Tabellenzeile heißt *‚Datensatz‘*. Einzelner Eintrag in Zeile ist das *‚Datum‘*. Kann leer sein, sollte er aber nicht.

Relationale Datenbanken können vollständig als Tabellen dargestellt werden.

Wichtigsten CRUD Datenbankoperationen: Create, Read, Update, Delete.

Es muss möglich sein, Daten

- zu erzeugen,
- zu lesen,
- zu ändern,
- zu löschen.

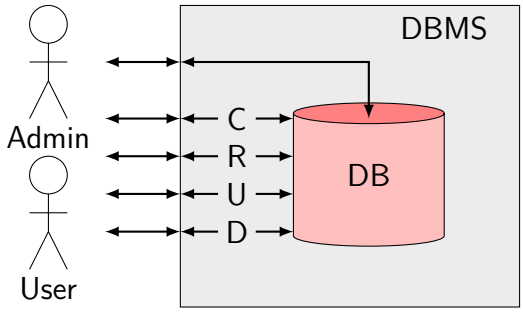
Später Umsetzung in SQL über

Create:	`INSERT INTO tab (col) VALUES (x)`
Read:	`SELECT * FROM tab WHERE col = x`
Update:	`UPDATE tab SET col = y WHERE col = x`
Delete:	`DELETE FROM tab WHERE col = x`

Datenbankmanagementsysteme

‚Datenbankmanagementsystem‘: (DBMS) ist die Gesamtheit aller Programme zur Erzeugung, Verwaltung und Manipulation einer Datenbank.

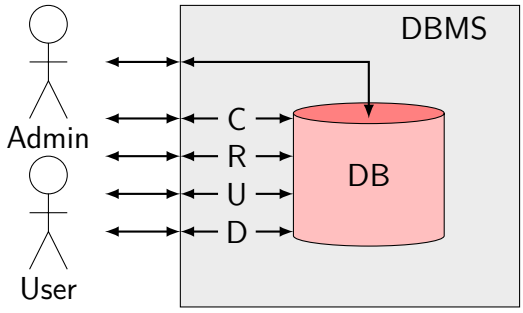
Zu Deutsch auch *‚Datenbankverwaltungssystem‘* (DBVS).



DBMS ‚versteckt‘ DB vor Anwendern und Administratoren. Bietet Schnittstellen für üblichen Zugriff auf DB und sinnvolle weitere Funktionalitäten.

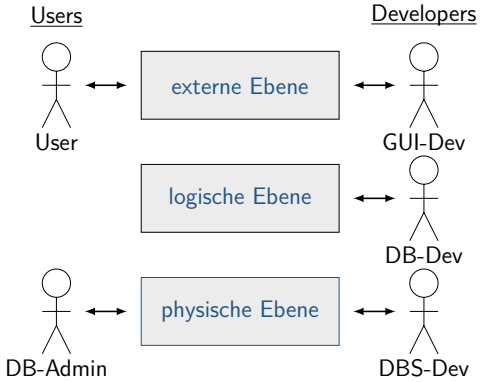
Sinvolle weitere Funktionalitäten:

- Abhängigkeiten zwischen Datenschematas/Tabellen bedenken und nutzen;
- Redundanz freie Datenhaltung und Integrität der Daten sichern;
- Parallele Nutzung koordinieren;
- Benutzerrechte verwalten;
- Daten sichern.



Ebenen eines DBMS

1. *‚externe Ebene'*
   Äußere Sicht auf die Datenbank. Wird verwendet für einfache Benutzer über GUI, für erfahrene Benutzer über Konsole.
2. *‚logische Ebene'*
   Beschreibt logischen Aufbau der Datenbank.
3. *‚physische Ebene'*
   Innere der Datenbank, in der Datenwerte physikalisch auf Festplatten gespeichert werden.



Wir bleiben in der externen Ebene.

Es gibt verschiedene Umsetzungen von Datenbankmanagementsystemen:

- RDBMS: relationale DBMS,
- ODBMS: objektorientierte DBMS,
- XDBMS: XML DBMS.
- ...

Relationale sind die üblichsten: **MySQL**, **PostgreSQL**, **Oracle**, **Sybase**, Microsoft Access, ...

Beispiele für ODBMS findet man unter **odbms.org**.

Beispiele für XDBMS sind **Sedna**, **BaseX**, ...

Wir verwenden das RDBMS **SQLite** mit graphischer Oberfläche **SQLiteCC** oder **SQLite Expert**. Möglicherweise nutzen wir auch noch die Einbettung in Python.

Datenbankentwurf

Entwicklungsprozess
Anforderungsanalyse & Miniwelt

Wollen DB entwickeln. Etwa für Comic-Sammlung, aus der Freunde ausleihen dürfen. Was ist dazu zu tun? Wie kommt man zu einer laufenden Anwendung?