Mensch-Computer-Interaktion

Bearbeitung zu *Grundlagen von Datenbanken*, WiSe 2016/17

Autor(en): Brigitte Kwasny

Inhaltsverzeichnis

Teil I
Organisatorisches

1 Ziele der Vorlesung

(a) Entwurf, Aufbau und Wartung von Datenbanken, insbesondere auf der Basis

(b) Sicherung der Ablaeufe auf Datenbanken *(Transaktionsprogramme)*

(c) Verwaltung und Handhabung semi-strukturierter Daten/Dokumente (XML)

(d) Entwicklung von (betrieblichen) Anwendungs- und Informationssystemen, insb. datenbankgestuetzter Anwendungen

(e) Nutzung (semi-)strukturierter Datenquellen unter Verwendung spezifischer Sprachansaetze

(f) Systemverantwortlicher fuer Informations- und Datenbanksysteme, insbesondere Unternehmens-, Datenbank-, Anwendungs- und Datensicherungsadministrator

2 Klausur

Die Klausur findet am 09.02.2017 von 12:30 - 14:30 Uhr im Audimax statt.

Teil II
Vorlesung 1

Heute werden wir von einer elektronischen Informationsflut "ueberrollt", deshalb gewinnen **Datenbankverwaltungssysteme** (DBMS - *database management systems*) eine immer groessere Bedeutung.
Ein DBMS besteht aus einer Menge von **Daten** und den zur Datenverarbeitung notwendigen Programmen:

(a) Die gespeicherten Daten werden oft als **Datenbasis** bezeichnet =>enthaelt die miteinander in Beziehung stehenden Informationseinheiten, die zur Kontrolle und Steuerung eines Aufgabenbereichs (evtl. eines ganzen Unternehmens) notwendig sind

(b) Die Gesamtheit der Programme zum Zugriff auf die Datenbasis, zur Kontrolle der Konsistenz und zur Modifikation der Daten wird als **Datenbankverwaltungssystem** bezeichnet

Oft werden diese Komponenten weniger scharf getrennt, sodass ein DBMS in der Regel beides meint.

3 Motivation, Einfuehrung und Grundbegriffe

3.1 Motivation

3.1.1 Motivation (1)

Betriebliche Informationssysteme und E-Business
Betriebliche Informationssysteme spiegeln die Geschaeftsmodelle von Unternehmen wider und dienen dazu, deren Arbeitslaeufe zu organisieren und zu unterstuetzen; sie sind **stark datenbankbasierte Anwendungen** mit **vielen Benutzern** und **transaktionsverarbeitende Systeme**. Dabei muss die **Integritaet der Daten** gewaehrleistet sein, sowie **hohen Durchsatz und kurze Antwortzeiten** muessen geschaffen werden. Sie sind **nicht nur Dialogsysteme**, sondern benoetigen meist einen **Batch**.

3.1.2 Motivation (2)

E-Business ist die Nutzung des Internets zu geschaeftlichen Zwecken aller Art. Bei E-Commerce fliesst Geld, denn es geht um Handel, also den Abschluss und die Abwicklung von Kaufvertraegen. Dabei werden Varianten unterschieden, je nachdem, wer mit wem handelt:
ein Unternehmen mit seinen Endkunden *(Business-to-Consumer, B2C)*, Unternehmen untereinander *(Business-to-Business, B2B)* oder Endkunden direkt miteinander ueber Boersen und Auktionen *(Consumer-to-Consumer, C2C)*.

3.1.3 Motivation (3)

E-Business braucht starke Softwaresysteme. Es sind **komplexe Systeme**, denn es genuegt nicht, sich mit einer gut gestalteten Web-Oberflaeche dem Benutzer zu praesentieren - werblich ansprechend, um ihn zu gewinnen; ergonomisch, um ihn nicht zu verlieren.
Dahinter muss mehr stehen: **eine flexible Anwendung**, die sich schnell an geaenderte Geschaeftsprozesse anpassen laesst, und eine **gehaltvolle Datenbank**.
E-Business-Systeme sind **eigentlich ueberbetriebliche Informationssysteme**, denn sie **verbinden** ueber ein Unternehmen hinausgehend Mitarbeiter, Lieferanten und Kunden und werden vor allem von Menschen genutzt.

3.1.4 Motivation (4)

Man muss eine noch nie da gewesene **Komplexitaet der Technologie** beherrschen. Man muss sich mit der **Programmierung der Web-Oberflaeche** auskennen *(HTML, XML, Java-Applets)*, **Netzprotokolle** *(z.B. HTTP)* und **Web-Server** einzusetzen verstehen, **Anwendungsprogramme in Java** schreiben und unter der **Transaktionskontrolle** von **Application-Servern** zum Laufen bringen.
Standard-Internet-Anwendungen *(z.B. Intershop)* sowie vorhandene (Legacy-)Systeme integrieren.

3.2 Miniwelt

Ein Datenbanksystem **verwaltet Daten** einer realen oder gedanklichen Anwendungswelt. Diese Daten gehen aus Informationen hervor, die stets aus den Sachverhalten und Vorgaengen dieser Anwendungswelt **durch gedankliche Abstraktionen** gewonnen werden.
Sie beziehen sich nur auf solche Aspekte des betrachteten Weltausschnitts, die fuer den Zweck der Anwendung relevant sind. Ein solcher Weltausschnitt wird auch als **Miniwelt** *(Diskurswelt)* bezeichnet.

TODO

3.3 Informationssystem

Definitionen nach Hansen:

(a) Ein **Informationssystem (IS)** besteht aus Menschen und Maschinen, die Informationen erzeugen und/oder benutzen und die durch Kommunikationsbeziehungen miteinander verbunden sind

(b) Ein **betriebliches IS** dient zur Abbildung der Leistungsprozesse und Austauschbeziehungen im Betrieb und zwischen dem Betrieb und seiner Umwelt

(c) Ein **rechnergestuetztes IS** ist ein System, bei dem die Erfassung, Speicherung und/oder Transformation von Informationen durch den Einsatz von EDV **teilweise automatisiert** ist =>KIS *(kooperatives Informationssystem)* besteht aus einer Menge unabhaengiger Systeme, die zusammen die angestrebte Leistung erbringen

(a) Eine **Datenbank (DB)**

- ist eine Sammlung gespeicherter operationaler Daten, die von den Anwendungssystemen eines Unternehmens benoetigt werden

(b) Ein **Datenbankverwaltungssystem (DBVS)**

- ist ein standardisiertes Softwaresystem zur Definition, Verwaltung, Verarbeitung und Auswertung der DB-Daten -item kann mittels geeigneter Parametrisierung an die speziellen Anwendungsbeduerfnissen angepasst werden *(hochgradig generisches System)*
- implementiert ein **Datenmodell**, das die Art der Datenstrukturen und generische Operationen zu deren Manipulation bereitstellt

(c) Ein **Datenbanksystem (DBS)**

- TODO

3.4 Zusammenfassung

Daten: objektive Welt der nicht-interpretierten Daten
Information: subjektive Welt der bewerteten Daten
grob: $DB + DBVS = DBS$; $DBS + AWS = KIS$
=> Heterogenitaet, Wachstum, Anforderungsvielfalt u.a. fuehren oft zu unabhaengigen IS, die zusammen als kooperatives IS die angestrebte Leistung erbringen muessen

Beispiele zu verschiedenen Informationssystemen: Universitaet, Produktionsbetriebes, Fluggesellschaft, Bank etc.

4 Anforderungen und (Schichten-)Modelle

Schichtenmodelle auch als **Beschreibungsmodell** bezeichnet.

4.1 Nutzung von Dateisystemen

Dateisystem:

(a) permanente Datenhaltung innerhalb von Betriebssystem-Dateien

(b) Betriebs/Dateisystem bietet Funktionen fuer die **Erzeugung / Loeschen** von Dateien, **Zugriffsmoeglichkeiten** auf Bloecke/Saetze der Datei und **einfache Operationen** zum Lesen/Aendern/Einfuegen/Loeschen von Saetzen

Daten auf Papier oder in isolierten Computer-Dateien abzulegen fuehrt meist zu folgenden schwerwiegenden Problemen:

(a) **Redundanz und Inkonsistenz**

- Bei Aenderungen von redundanten Daten kann dies zu Inkonsistenzen fuehren, wenn nur eine Kopie der Daten geaendert wird, die andere aber noch im vealteten Zustand beibehalten wird

(b) **Beschraenkte Zugriffsmoeglichkeiten**

- die in isolierten Dateien abgelegten Daten kann man unmoeglich miteinander "verknuepfen"

(c) **Probleme des Mehrbenutzerbetriebs**

- Die heutigen Dateisysteme bieten entweder gar keine oder nur sehr rudimentaere Kontrollmechanismen fuer den Mehrbenutzerbetrieb. Daten werden i.A. von sehr vielen Anwendern innerhalb und ausserhalb der jeweiligen Organisation genutzt *(z.B. Flugreservierungssystem)*. **Bei unkontrolliertem Zugriff** kann es sehr leicht zu unerwuenschten Anomalien kommen *(z.B. Gleichzeitiges Editieren desselben Datensatzes durch zwei Benutzer)* =>**"lost update"**

(d) **Verlust von Daten**

- Wenn Daten in isolierten Dateien gehalten werden, wird die Wiederherstellung eines konsistenten - d.h. eines gemaess der realen Welt gueltigen Zustands der Gesamtinformationsmenge im Fehlerfall sehr schwierig. Im Allgemeinen bieten Dateisysteme bestenfalls die Moeglichkeit einer periodisch durchgefuehrten Sicherung der Dateien. **Datenverluste**, die waehrend der Bearbeitung von Dateien oder nach der letzten Sicherungskopie auftreten, sind i.A. nicht auszuschliessen

(e) **Integritaetsverletzung**

- Je nach Anwendungsgebiet gibt es vielfaeltigste, sich global ueber mehrere Informationseinheiten erstreckende Integritaetsbedingungen *(z.B. Stundenten muessen erst die Pflichtseminare abgeschlossen haben, bevor sie zur Pruefung zugelassen werden oder max zwei Pruefungsversuche)*. Die Einhaltung derartiger Integritaetsbedingungen ist bei der isolierten Speicherung der Informationseinheiten in verschiedenen Dateien sehr schwierig, da man zur Kontrolle Daten aus unterschiedlichen Dateien verknuepfen muss *(siehe oben)*. Ausserdem will man im Allgemeinen nicht nur die **Konsistenzbedingungen** ueberpruefen, sondern die **Einhaltung erzwingen**

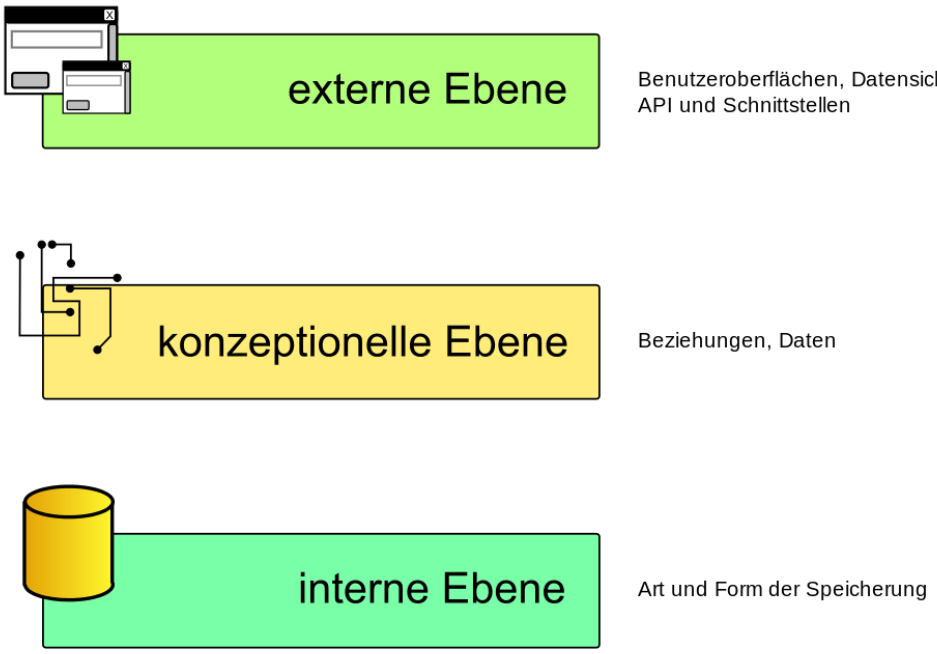
(f) **Sicherheitsprobleme**

- Nicht alle Benutzer sollten Zugriff auf die gesamten gespeicherten Daten haben. Und erst recht sollten nur bestimmte ausgewaehlte Benutzer das **Privileg zum Editieren** haben

(g) **Hohe Entwicklungskosten**

- In vielen Faellen muss fuer die Entwicklung eines neuen Anwendungsprogrammes praktisch "das Rad neu erfunden" werden. Jedesmal muss sich der Anwendungsprogrammierer zusaetzlich zu Fragen der Dateiverwaltung mit zumindest einer Teilmenge der obigen Probleme auseinandersetzen.

Alle Probleme, die hier aufgezaehlt wurden, werden durch ein DBMS abgeloest, da es interne Moeglichkeiten fuer die Realisierung bietet.



4.2 Datenabstraktion / Schema-Architektur (Folie)

Man unterscheidet drei Abstraktionsebenen *(ANSI-SPARC-Architektur oder auch Drei-Schema-Architektur)* im Datenbanksystem:

(a) **Die physische / interne Ebene**

- Auf dieser Ebene wird festgelegt, wie die Daten gespeichert sind. Im Allgemeinen sind Daten auf dem Hintergrundspeicher *(Festplatte)* abgelegt

(b) **Die logische / konzeptionelle Ebene**

- Auf der logischen Ebene wird in einem sogenannten **Datenbankschema** festgelegt, welche Daten abgespeichert werden

(c) **Die Sichten / Externe Ebene**

- Waehrend das Datenbankschema der logischen Ebene ein integriertes Modell der gesamten Informationsmenge des jeweiligen Anwendungsbereichs darstellt, werden **in den Sichten** Teilmengen der Information bereitgestellt =>Die Sichten sind **auf die Beduerfnisse** der jeweiligen Benutzer(-gruppen) zugeschnitten *(z.B. drei Studenten, die Professoren oder der Hausmeister)* =>Zugriffsschutz, Reduktion der Komplexitaet

Zum einen wollen Anwender i.A. nur einen Ausschnitt des gesamten Informationsmodells zu sehen bekommen. Zum anderen koennen bestimmte kritische Informationsteile in den Sichten ausgeblendet werden, um dadurch den Datenschutz zu gewaehrleisten.

4.3 Datenunabhaengigkeit

nicht in VL erwaehnt

Die drei Ebenen eines DBMS gewaehrleisten einen bestimmten Grad der **Datenunabhaengigkeit**. Dies ist analog zum Konzept der **abstrakten Datentypen** (ADTs) in Programmiersprachen. Durch eine wohldefinierte Schnittstelle wird die "darunterliegende" Implementierung verdeckt, so dass man - bei Beibehaltung der Schnittstelle - die Realisierung variieren kann, ohne dass die Benutzer der Schnittstelle davon in Mitleidenschaft gezogen werden.

Aufgrund der drei Schichten ergeben sich zwei Stufen der Datenunabhaengigkeit im DBMS:

(a) **Physische Datenunabhaengigkeit**

- TODO

(b) **Logische Datenunabhaengigkeit**

- TODO

Die heutigen Datenbanksysteme erfuellen zumeist die physische Datenunabhaengigkeit. Die logische Datenunabhaengigkeit kann schon rein konzeptuell nur fuer einfachste Modifikationen des Datenbankschemas gewaehrleistet sein.

4.4 Grundbegriffe

Datenbank (DB) als **Abbildung einer Miniwelt**

(a) Vorgaenge und Sachverhalte werden als gedankliche Abstraktionen der Miniwelt erfasst und als Daten **in** der Datenbank gespeichert

(b) Daten beziehen sich nur auf solche Aspekte der Miniwelt, die fuer die Zwecke der Anwendung relevant sind

(c) Eine DB ist integritaetserhaltend (bedeutungstreu), wenn ihre Objekte Modelle einer gegebenen Miniwelt repraesentieren

Datenmodell, DB-Schema und DB-Instanz
Datenbankverwaltungssysteme basieren auf einem **Datenmodell**, das sozusagen die Infrastruktur fuer die Modellierung der realen Welt zur Verfuegung stellt. Das Datenmodell legt die Modellierungskonstrukte fest, mittels derer man ein computerisiertes Informationsabbild der realen Welt *(bzw. des relevanten Ausschnitts)* generieren kann.

(a) Datenmodell legen Regeln fest, nach denen die Objekte von DBs erzeugt und veraendert werden

(b) DB-Schema legt die Auspraegungen der Objekte fest, welche die DB fuer eine bestimmte Miniwelt einnehmen kann

(c) DB-Instanz entspricht tatsaechlicher (schema-konformer) Auspraegung einer Menge von Objekten

Das Datenmodell ist somit analog zu einer Programmiersprache: Es legt die generischen Strukturen und Operatoren fest, die man zur Modellierung einer bestimmten Anwendung ausnutzen kann. Eine Programmiersprache legt die Typkonstruktoren und Sprachkonstrukte fest, mit deren Hilfe man spezifische Anwendungsprogramme realisiert.

Das Datenmodell besteht demnach aus zwei Teilsprachen:

(a) der **Definitionssprache** *(DDL - data definition language)*

(b) der **Datenmanipulationssprache** *(DML - data manipulation language)*

Die DDL wird benutzt, um die Struktur der abzuspeichernden Datenobjekte zu beschreiben. Dabei werden gleichartige Datenobjekte durch ein gemeinsames Schema *(analog zu einem Datentyp in Programmiersprachen)* beschrieben. Die Strukturbeschreibung aller Datenobjekte des betrachteten Anwendungsbereichs nennt man das **Datenbankschema**.

Die Datenmanipulationssprache besteht aus

- der **Anfragesprache** *(Query Language)* und
- der "eigentlichen" Datenmanipulationssprache zur Aenderung von abgespeicherten Daten und zum Loeschen von gespeicherten Daten

Die DML (inkl. Anfragesprache) kann in zwei unterschiedlichen Arten genutzt werden:

- **interaktiv**, TODO
- TODO

Beschreibung und Handhabung der Daten

(a) Daten muessen interpretierbar sein

(b) Daten muessen bei allen am Austausch beteiligten Partnern (Systemen, Komponenten) die Ableitung derselben Information erlauben

(c) Einsatzspektrum verlangt generische Vorgehensweise

(i) Beschreibung der zulaessigen DB-Zustaende

(ii) Beschreibung der zulaessigen Zustandsuebergaenge (generische Operatoren)

Anwendungsprogrammier-Schnittstelle (API)

(a) Operatoren zur Definition von Objekttypen *(Beschreibung der Objekte)*

(i) **DB-Schema:** Welche Objekte sollen in der DB gespeichert werden?

(b) Operatoren zum Aufsuchen und Veraendern von Daten

(i) **AW-Schnittstelle:** Wie erzeugt, aktualisiert und findet man DB-Objekte?

(c) Operatoren zur Definition von Integritaetsbedingungen *(Constraints)*

(i) **Sicherung der Qualitaet:** Was ist ein akzeptabler DB-Zustand?

(d) Operatoren zur Definition von Zugriffskontrollbedingungen

(i) **Massnahmen zum Datenschutz:** Wer darf was?

4.5 Datenbankschema und Auspraegung

nicht in VL erwaehnt

TODO

4.6 Einordnung der Datenmodelle

nicht in VL erwaehnt

In Abbildung 1.2 sind die grundlegendsten Phasen der Datenmodellierung gezeigt.

Modelle des konzeptuellen Entwurfs
Man beginnt beim Datenbankentwurf mit der Abgrenzung eines Teils der "realen Welt", um den Ausschnitt *(Miniwelt)* zu bestimmen, der in der Datenbank modelliert werden soll. Diese Miniwelt wird dann konzeptuell modelliert.

Fuer die konzeptuelle Modellierung gibt es mehrere moegliche Datenmodelle:

(a) Entity-Relationship-Modell *(auch Gegenstand-Beziehungs-Modell)*

(b) semantisches Datenmodell

(c) objektorientierte Entwurfsmodelle *(wie UML)*

4.7 Anforderungen an ein DBS

(a) **Kontrolle ueber die operationalen Daten**

(b) **Leichte Handhabbarkeit der Daten**

(c) **Kontrolle der Datenintegritaet**

(d) **Leistung und Skalierbarkeit**

(e) **Hoher Grad an Daten-Unabhaengigkeit**

TODO

4.8 Schichtenmodell fuer DBS

TODO

4.9 Dynamischer Ablauf einer DB-Operation

`SELECT * FROM 'person' WHERE pnr = '12345'`

(a) Vervollstaendigen der Verarbeitungsinformation aus Konzeptionellem und Internem Schema

  (i) Ermittlung der Seiten *(z.B. durch Hashing)*

(b) Zugriff auf DB-Puffer: falls erfolgreich, dann weiter mit 7

(c) Zugriff auf DB ueber DB-Pufferverwaltung/Betriebssystem

(d) Durchfuehrung des E/A-Auftrages

(e) Ablegen der Seite im DB-Puffer

(f) Uebertragen des Anfrageergebnisses in den Arbeitsbereich der Anwendung

(g) Statusinformation: Return-Code, Cursor-Info

(h) Manipulation mit Anweisungen der Programmiersprache

4.10 Zusammenfassung

DBS-Charakteristika:
Zentralisierte Verwaltung der operationalen Daten *(Rolle des DBA)*, Adaequate Schnittstellen *(Datenmodell und DB-Sprache)*, Datenkontrolle - insb. zentrale Kontrolle der Datenintegritaet und kontrollierter Mehrbenutzerbetrieb, Leistung und Skalierbarkeit, Hoher Grad an Daten-Unabhaengigkeit

Beschreibungsmodelle fuer ein DBS:
Schichtenmodell, Drei-Schema-Architektur

Programmierschnittstellen: *(siehe Schichtenmodell)*
mengenorientierte DB-Schnittstelle *(relationale DBS)*, satzorientierte DB-Schnittstelle *(hierarchische und netzwerkartige DBS)*, interne Satzschnittstelle *(zugriffsmethodenorientierte Programmierschnittstelle)*

5 Zusammenfassung: Vorlesung 1

TODO

Vorlesung 2

Das mit Abstand am haeufigsten benutzte Modell fuer den konzeptuellen Entwurf ist das **Entity-Relationship-Modell**. In der konzeptuellen Entwurfsphase werden die in der realen Welt vorkommenden Konzepte in Gegenstandsmengen und Beziehungen zwischen diesen Gegenstandsmengen strukturiert.

Abbildung 1.3 zeigt diese "intellektuelle" Aufgabe fuer einen ganz kleinen Ausschnitt der Universitaetswelt. Es werden die Gegenstandsmengen *Studenten, Professoren* und *Vorlesungen* ermittelt. Weiterhin werden die Beziehungen *hoeren* und *lesen* bestimmt.

Die konzeptuellen Datenmodelle verfuegen im Allgemeinen nur ueber eine DDL und haben keine Datenmanipulationssprache, da sie nur die Struktur der Daten beschreiben. Sie verzichten auf die Abbildung von individuellen Datenobjekten, d.h. es werden keine Datenbankauspraegungen erzeugt. Deshalb benoetigen sie keine Datenmodifikationssprache (DML).

6 Informationsmodellierung

6.1 DB-Entwurf und Modellierung

Ziel: Modellierung einer Miniwelt: modellhafte Abbildung eines anwendungsorientierten Ausschnitts der realen Welt (Miniwelt) und Nachbildung von Vorgaengen durch **Transaktionen**
Nebenbedingungen: genaue Abbildung, hoher Grad an Aktualitaet, Verstaendlichkeit, Natuerlichkeit, Einfachheit
Zwischenziel: Erhebung der Information in der Systemanalyse (Informationsbedarf), **Informationsmodell** (allg. Systemmodell)
Bestandteile: Objekte: Entities, Beziehungen: Relationships

Informationsmodell: Darstellungselemente und Regeln, eine Art formale Sprache, um Informationen zu beschreiben
Informationen ueber Objekte und Beziehungen nur, wenn: unterscheid- und identifizierbar, relevant und selektiv beschreibbar

6.2 Entity-Relationship-Modell

Modellierungskonzepte: Entity-Mengen (Objektmengen), Wertebereiche, Attribute, Primaerschluessel, Relationship-Mengen (Beziehungsmengen)
Klassifikation der Beziehungstypen: benutzerdefinierte Beziehungen, Abbildungstyp *(*$1:1$, $1:n$, $n:m$*)*
=>**Ziel:** Festlegung von semantischen Aspekten, explizite Definition von strukturellen Integritaetsbedingungen

(!) Das ERM modelliert die **Typ-**, nicht die Instanz**ebene**; es macht also Aussagen ueber Entity- und Relationship-Mengen, nicht jedoch ueber einzelne ihrer Elemente (Auspraegungen). Die Modellierungskonzepte des ERM sind haeufig zu ungenau oder unvollstaendig. Sie **muessen** deshalb **ergaenzt werden** durch **Integritaetsbedingungen** (Constraints).

6.2.1 Entity

Entities sind wohlunterscheidbare Dinge der Miniwelt (Diskurswelt) und besitzen Eigenschaften, deren konkrete Auspraegungen als Werte bezeichnet werden.

Entity-Mengen: Zusammenfassung von "aehnlichen" oder "vergleichbaren" Entities und haben gemeinsame Eigenschaften
=>Bsp: Abteilungen, Angestellte, Projekte ... / Buecher, Autoren, Leser ...

Wertebereiche und Attribute: (1) Die moeglichen oder "zulaessigen" Werte fuer eine Eigenschaft nennen wir Wertebereich (oder Domain). **(2)** Die (bei allen Entities einer Entity-Menge auftretenden) Eigenschaften werden als Attribute bezeichnet und **(3)** ein Attribut ordnet jedem Entity einer Entity-Menge einen Wert aus einem bestimmten Wertebereich (dem des Attributs) zu.

6.2.2 Beispiel: "Buch" (Entity-Typ)

PICTURE in Folie

- jedem Attribut ist einem geeignetem Wertebereich zugeordnet
- Name der Entity-Menge sowie die zugehoerigen Attribute sind **zeitinvariant**
- Entity-Menge und ihre Entities sind **zeitveraenderlich**

PICTURE in Folie
Erhoehung der Modellierungsgenauigkeit durch

- einwertige Attribute
- mehrwertige Attribute (Doppelovale)
- zusammengesetzte Attribute (hierarchisch angeordnete Ovale)
- Verschachtelungen sind moeglich

6.2.3 Wie wird ein Entity identifiziert?

Entities muessen "wohlunterscheibar" sein und die Information ueber ein Entity ist ausschliesslich durch den (Attribut-)Wert definiert.
Identifikation eines Entities durch Attribut:

- $1:1$ Beziehung (ggf. kuenstlich erzwungen durch laufende Nummer)

TODO mit Formeln

6.2.4 Definition: Entity-Typ

Ein Entity-Typ hat die Form $E = (X, K)$ mit einem Namen E, einem Format X und einem Primaerschluessel K, der aus (einwertigen) Elementen von X besteht. Die Elemente eines Formats X werden dabei wie folgt beschrieben:

- einwertige Attribute A
- mehrwertige Attribute A
- zusammengesetzte Attribute $A(B_1, ..., B_k)$

6.2.5 Definition: Wertebereich / Domain

Sei $E = (X, K)$ ein Entity-Typ und $attr(E)$ die Menge aller in X vorkommenen Attributnamen. Jedem $A \in attr(E)$, das nicht einer Zusammensetzung voransteht, sei ein Wertebereich $W(A)$ zugeordnet. Fuer jedes $A \in attr(E)$ sei

- $dom(A) := W(A)$, falls A einwertig
- $dom(A) := 2^{W(A)}$, falls A mehrwertig
- $dom(A) := W(B_1)x...xW(B?k)$, falls A aus einwertigen $B_1, ..., B_k$ zusammengesetzt

Besteht A aus mehrwertigen oder zusammengesetzten Attributen, wird die Definition rekursiv angewendet.

6.2.6 Definition: Entity und Entity-Menge

Sei $E = (X, K)$ ein Entity-Typ mit $X = (A_1, ..., A_m)$. A_i sei $dom(A_i)(1 \leq i \leq m)$ zugeordnet.

- Ein Entity e ist ein Element des Kartesischen Produkts aller Domains, d.h. $e \in dom(A_1)x...xdom(A_m)$
- Eine Entity-Menge E^t (zum Zeitpunkt t) ist eine Menge von Entities, welche K erfuellt, d.h. $E^t \subseteq dom(A_1)x...xdom(A_m)$

E^t wird auch als der Inhalt bzw. der aktuelle Wert (Instanz) des Typs E zur Zeit t berechnet.

6.2.7 Definition Relationship, Relationship-Typ und Relationship-Menge

Ein Relationship-Typ hat die Form $R = (Ent, Y)$. Dabei ist R der Name des Typs, Ent bezeichnet die Folge der Namen der Entity-Typen, zwischen denen die Beziehung definiert ist und Y ist eine (moeglicherweise leere) Folge von Attributen der Beziehung.

Sei $Ent = (E_1, ..., E_k)$ und fuer ein beliebiges, aber festes t sei ${E_i}^t$ der Inhalt des Entity-Typs E_i, $1 \leq i \leq k$. Ferner sei $Y = (B_1, ..., B_n)$. Ein Relationship r ist ein Element des Kartesischen Produktes aus allen ${E_i}^t$ und den Domains der B_j, d.h.

- $r \in {E_i}^t x ... x {E_i}^t x dom(B_1) x ... x dom(B_n)$ bzw
- $r = (e_1, ..., e_k, b_1, ..., b_n)$ mit
- $e_i \in {E_i}^t$ fuer $1 \leq i \leq k$ und $b_j \in dom(B_j)$ fuer $1 \leq j \leq n$

Eine Relationship-Menge R^t (zur Zeit t) ist eine Menge von Relationships, d.h.

- $R^t \subseteq {E_i}^t x ... x {E_i}^t x dom(B_1) x ... x dom(B_n)$

Eigenschaften von Relationship-Mengen: Grad n der Beziehung *(degree)*, gewoehnlich $n = 2$ oder $n = 3$, Existenzabhaengigkeit, Beziehungstyp *(connectivity)* und Kardinalitaet

PICTURE von Folie
Ausleihe = ((Leser, Buch), (RDatum))
Eigenschaften: Grad = 2, Existenzabhaengigkeit = false, Beziehungstyp = $n : m$

6.2.8 Beispiele

TODO

6.3 Erweiterungen des ERM

TODO

6.4 Zusammenfassung

DB-Entwurf umfasst: (1) Informationsbedarfsanalyse, **(2)** konzeptionelles DB-Schema (->Informationsmodell), **(3)** logisches DB-Schema *(nicht diskutiert)*, **(4)** physisches DB-Schema *(nicht diskutiert)*

ERM-Charakteristika: (1) Modellierung bezieht sich auf die Typeebene, **(2)** relevante Zusammenhaenge der Miniwelt werden durch Entity- und Relationship-Mengen modelliert; sie werden genauer durch Attribute, Wertebereiche, Primaerschluessel/Schluesselkandidaten beschrieben, **(3)** Klassifikation von Beziehungstypen dient der Spezifikation von strukturellen Integritaetsbedingungen, **(4)** anschauliche Entwurfsdarstellung durch ER-Diagramme und **(5)** relativ karges Informationsmodell

Einfuehrung weiterer Modellierungskonzepte: TODO

Vorlesung 3