

Entity Relationship Modelle (ERM)

8.1 Einführung




- Entity-Relationship-Modell (ER-Modell, ERM)
 - Anwendungsbereich
 - ERDs werden im Bereich des Datenbank-Entwurfs eingesetzt.
 - Ziel
 - Beschreibung permanent gespeicherter Daten und ihrer Beziehungen untereinander, d.h. Beschreibung der statischen Zusammenhänge von datenbank-intensiven Anwendungen
 - Verhalten wird nicht modelliert!
 - Phase
 - Auf Grund des großen Verbreitungsgrades werden ERMs oft schon in der Spezifikation eingesetzt (macht ausschließlich bei datenbank-intensiven Anwendungen Sinn).
 - Darstellung
 - ER-Modelle werden durch Entity Relationship Diagramme (ERD) repräsentiert

[Chen, P.: "The entity-relationship model: toward a unified view of data", ACM Transactions on Database Systems 1:1, pp 9-36, 1976]

8.1 Einführung

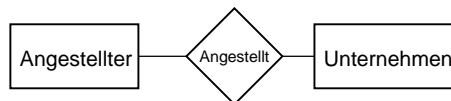
- Entity-Relationship-Modell ff.
 - Semantische Datenmodellierung
 - ERM um Aggregation und Vererbung erweitert

8.1 Einführung

- Grundelemente
 - Entitätstypen (ungenau bezeichnet als Entities)

 - Relationship-Typen (ungenau bezeichnet als Relationships)

 - Attributtypen (ungenau bezeichnet als Attribute)


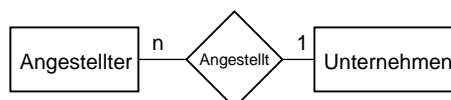
8.1 Einführung

- Bedeutung der Grundelemente
 - Entitätstypen
 - Bezeichnen eine Menge strukturgleicher Entitäten (Objekte). Dies ist eine Klassifizierung
 - Entitätstyp entspricht UML-Klasse, Entität UML-Objekt
 - Relationship-Typ zwischen zwei (oder mehreren) Entitätstypen
 - Bedeutet, dass Entitäten der verbundenen Entitätstypen in einer Beziehung zueinander stehen (in manchen Notationen werden die Relationship-Typen (d.h. das Rautensymbol) weggelassen und nur Entitätstypen verwendet).
 - Entspricht der Assoziation aus UML.



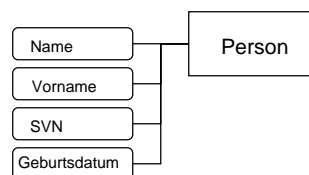
8.1 Einführung

- Bedeutung der Grundelemente ff.
 - Verbindung zwischen einem Relationship-Typen und einem Entitätstypen
 - Bedeutet, dass Entitäten des Entitätstypen an Beziehungen, die durch den Relationship-Typen definiert werden, teilnehmen. Eine solche Verbindung hat eine Stelligkeit, die angibt, wieviele Entitäten teilnehmen müssen / können.



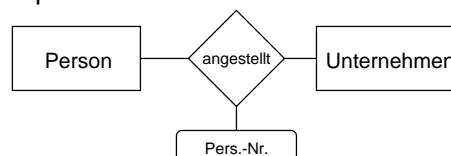
8.1 Einführung

- Bedeutung der Grundelemente ff.
 - Ein Attributtyp eines Entitätstyps
 - Definiert ein Merkmal, das die Entitäten des Entitätstyps aufweisen. Attributtypen werden durch einen Namen und einen Wertebereich definiert.
 - Der Entitätstyp Person hat die Attributtypen Name (Typ String), Vorname (Typ String), Geburtsdatum (Typ Datum), Sozialversicherungsnummer (Typ integer).
 - Entspricht Attributen von UML-Klassen



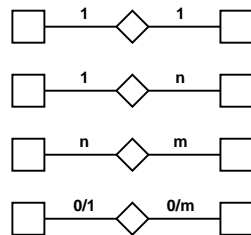
8.1 Einführung

- Bedeutung der Grundelemente ff.
 - Ein Attributtyp eines Relationship-Typen
 - Beschreibt ein Merkmal, das die Beziehung zwischen zwei (oder mehreren) Entitätstypen aufweist.
 - Der Relationship-Typ „angestellt“ zwischen „Unternehmen“ und „Person“ hat den Attributtypen „Personalnummer“ (Typ integer).
 - Der Relationship-Typ „Mietverhältnis“ zwischen „Nutzungseinheit“, „Mieter“ und „Vermieter“ hat den Attributtypen „Beginn“ (Typ Datum).
 - Entspricht assoziativen Klassen in UML



8.1 Einführung

- Kardinalität von Relationship-Typen

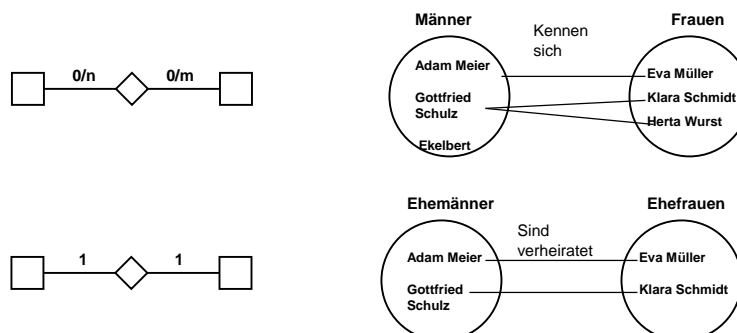


Durch die 0 werden
kann-Verbindungen beschrieben
(Erweiterung der Chen-Notation)

- Die Stelligkeit zwischen einem Entitätstypen A und einem Relationship-Typen R gibt an, in wievielen Beziehungen des Typs R eine Entität vom Typ A teilnehmen kann.

8.1 Einführung

- Kardinalitäten und Ausprägungen



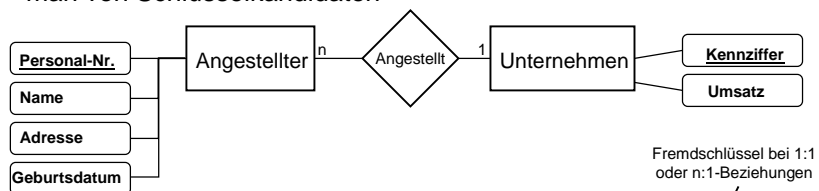
8.1 Einführung

- Alternative Darstellungsformen von Kardinalitätsangaben

Chen-Notation	Numerische Notation(min,max)	Krähenfußnotation	Pfeilnotation	Bachmannotation
	(0,1)			
	(1,1)			
	(0,n)			
	(1,n)			

8.2 Schlüssel und Tabellen

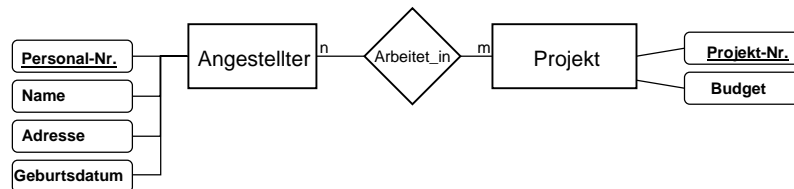
- Attribute als Schlüssel
 - Identifizieren Entitäten
 - Daher sog. identifizierende Attribute (\neq beschreibende Attribute)
 - Bei mehreren Schlüsseln einen Primärschlüssel auszeichnen
 - Schlüssel sind stets minimale Attributkombinationen, sonst spricht man von Schlüsselkandidaten



Angestellter	<u>Personal-Nr.</u>	Name	Adresse	Geburtsdatum	<u>Kennziffer</u>
	1999	Müller	Müllerstrasse 2, ..	01.01.1950	50
	2000	Maier	Maierstrasse 10, ..	10.05.1963	21

8.2 Schlüssel und Tabellen

- Attribute als Schlüssel
 - Bei m:n-Beziehungen dritte Tabelle mit Primärschlüsseln der beiden ersten Tabellen



Arbeitet_in	Personal-Nr.	Projekt-Nr.
	1999	101
	2000	321

8.2 Schlüssel und Tabellen

- Verknüpfung von Daten und Funktionen
 - Sind bereits Funktionen identifiziert (z.B. per Funktionsbaum), können diese per Assoziationsmatrix mit den Daten verknüpft werden

Funktion/Tabelle	Angestellter	Unternehmen	Projekt	angestellt	Arbeitet_in
Stammdaten anlegen	c				
Stammdaten aktualisieren	u				
Unternehmensdaten anlegen		c			
Projektdaten anlegen			c		
Projekt löschen			d		
Gehaltsabrechnung erstellen	r	r		r	
Unternehmensbilanz erstellen	r	r	r	r	r

c: create, u: update, r: read, d: delete

8.3 Semantische Datenmodelle

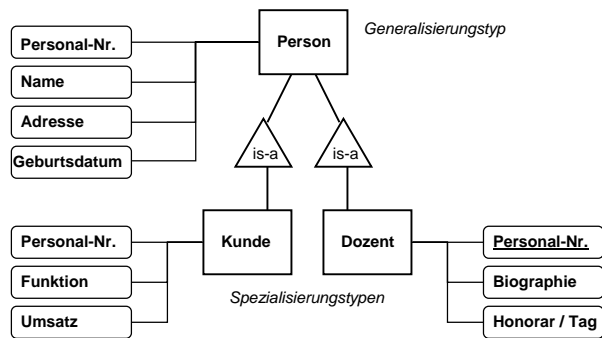
- Strukturierung von ERDs
 - Klassifikation
 - Generalisierung
 - (nicht im Standard ERD, aber in fast allen gängigen Erweiterungen)
 - Aggregation
 - (nicht im Standard ERD, aber in fast allen gängigen Erweiterungen)
 - Assoziation
 - (rudimentär durch Relationship-Typen, komplizierte Assoziationen können nicht dargestellt werden)

8.3 Semantische Datenmodelle

- Generalisierung/Spezialisierung
 - Vererbung von Attributen (ähnlich der Vererbung bei objektorientierten Programmiersprachen)
 - „ist ein/is-a“ Beziehung
 - Beispiel:
 - Der Entitätstyp „Dozent“ ist eine Spezialisierung des Entitätstyps „Person“ (Generalisierung)

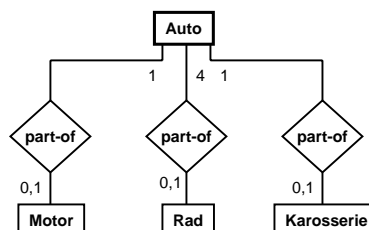
8.3 Semantische Datenmodelle

- Beispiel einer Generalisierungshierarchie



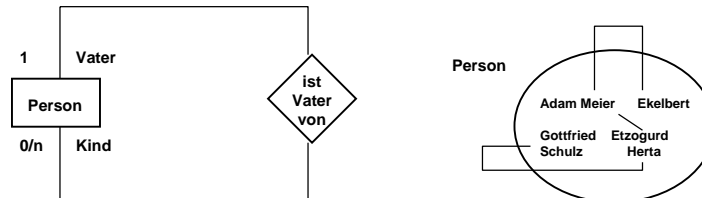
8.3 Semantische Datenmodelle

- Beispiel für eine Aggregation



8.3 Semantische Datenmodelle

- Rekursive Relationship-Typen
 - Ziel und Quelle sind der gleiche Entitätstyp



- Entspricht reflexiver Assoziation in UML

8.4 Unternehmensdatenmodelle

- Unternehmensdatenmodelle
 - Beschreibung sämtlicher Informationsstrukturen und Geschäftsprozesse eines Unternehmens unter Berücksichtigung der Schnittstellen untereinander
 - Ziel: Vermeidung von Brüchen oder Redundanzen bei übergreifenden Geschäftsprozessen
 - Beispiel: Strategisches Datenmodell mit den wichtigsten Entitätstypen, Beziehungen, Aggregationen und Generalisierungen
 - Aber
 - Aufwand zur Erstellung eines Unternehmensdatenmodells ist enorm groß und kostenintensiv. Eine Wirtschaftlichkeitsbetrachtung ist immer vorzunehmen. (Faustregel: "Die ersten 80% kosten soviel Geld und Zeit wie die letzten 20%")
 - Die Unternehmensdaten sind unterschiedlich zeitstabil. Ein Unternehmensdatenmodell sollte vor allem diejenigen Entitäts-Typen und -Beziehungen abbilden, die sich nicht oder nur sehr selten ändern.

8.5 Bewertung von ERDs

- Regeln / Plausibilitätsfragen für ERDs
 - Hat jeder Entitätstyp mindestens ein Attribut?
 - Sonst kein Entitätstyp
 - Entitätstypen durch Substantive, Relationship-Typen durch Verben beschrieben?
 - Sind Mehrfach-IS-A-Beziehungen wirklich nötig?
 - Benutzen Spezialisierungstypen alle attribute des Generalisierungstyps?
 - Gibt es unterschiedliche Abstraktionsebenen, die in einem ERD auftauchen?
 - Können 1:1 Beziehungen zusammengefasst werden?
 - Wo endet das Modell, wo fängt es an, was passiert außerhalb des modellierten Ausschnitts?
 - Was sind inhaltliche Beschreibungen? Gibt es vorweggenommene Entwurfsentscheidungen?

8.5 Bewertung von ERDs

- Beispiel
 - Fußballverein
 - Profiverein
 - Amateurverein
 - Erstligaverein
 - Zweitligaverein
 - Bundesligaspiel
 - DFB -Pokalspiel
 - Umsatz
 - erreichte Punkte
 - Tabelle
 - Schiedsrichter
 - Präsidenten

8.6 Fazit

- Semantische Datenmodellierung ist für kaufmännischen Anwendungsbereich wichtig
- Semantische Datenmodellierung ist Voraussetzung für relationalen DB-Entwurf
- Dynamische Aspekte können allerdings nicht dargestellt werden
- Unübersichtlich, falls sehr umfangreiche Datenmodelle. Kein Verfeinerungsmechanismus verfügbar.

8.7 Literatur

- [Chen76] Chen, P.: "The entity-relationship model: toward a unified view of data", ACM Transactions on Database Systems 1:1, pp 9-36, 1976
- [Dat90] Date, C. J.: An Introduction to Database Systems. Vol. I, 5th edition, Addison-Wesley, Reading (Mass.), 1990
- [EN94] Elmasri, R./ Navathe, S.B.: Fundamentals of Database Systems, 2nd edition. Benjamin/Cummings, Redwood City (Cal.) 1994
- [Ull88] Ullman, J.D.: Principles of Database and Knowledge-Base Systems. Vol I, Computer Science Press, Rockville (Md.) 1988
- [Vos94] Vossen, G.: Datenmodelle, Datenbanksprachen und Datenbank-Management-Systeme, 2. Auflage. Addison-Wesley, Bonn 1994
- [Bal00] Balzert, H.: Lehrbuch der Software-Technik, Band 1, 2. Auflage, Spektrum, 2000
- [Sch90] Scheer, A.-W.: Unternehmensdatenmodell. In: Lexikon der Wirtschaftsinformatik, Springer, 1990, S. 438-440.

Datenflussorientierte Spezifikation mit Datenflussdiagrammen (DFDs)

Agenda

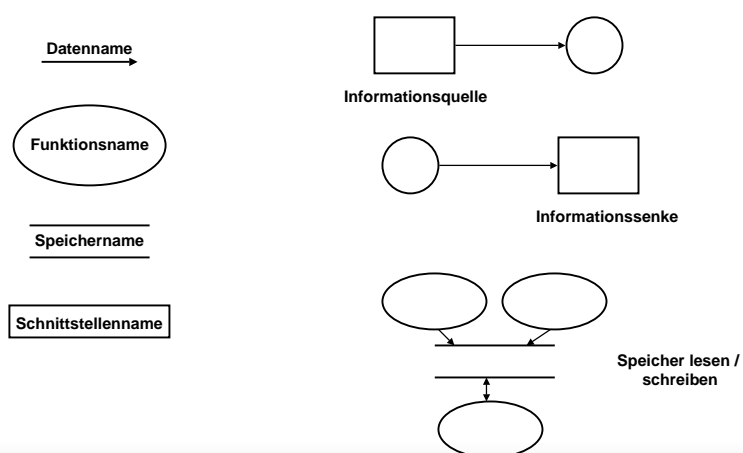
- 9 Datenflussdiagramme (DFDs)
 - 9.1 Einführung
 - 9.2 Fazit
 - 9.3 Literatur

9.1 Einführung

- Datenflussdiagramme
 - Beschreiben
 - den Fluss von Daten zwischen Funktionen, Speichern und Schnittstellen
 - die Transformation von Daten durch Funktionen
 - Abstrahieren weitgehend von Kontrollflussaspekten (welche Aktivität wird wann ausgeführt)
 - Sind die Basis von Sprachen, die eng mit Methoden und Werkzeugen verknüpft sind (SA, SADT, SA/RT).
 - Verbreiteste Darstellungsweise von DeMarco
 - [DeM79: DeMarco, T.: Structured Analysis and System Specification, Yourdon Press, 1979]
 - Im folgenden beschränken wir uns auf die Grundelemente von Datenflussdiagrammen.

9.1 Einführung

- Syntax von Datenflussdiagrammen

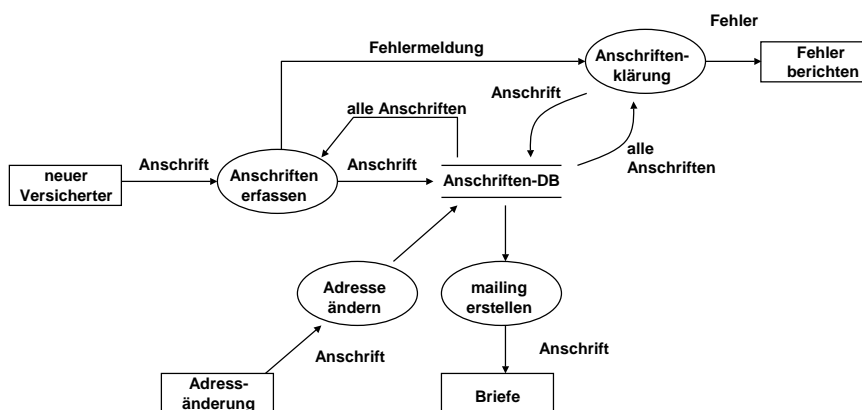


9.1 Einführung

- Idee
 - Das System läuft bereits (Initialisierung und Terminierung irrelevant)
 - Informationen entstehen in Informationsquellen und fließen zu Funktionen
 - Funktionen transformieren ankommende Datenflüsse in ausgehende Datenflüsse
 - Speicher dienen zur Ablage von Informationen

9.1 Einführung

- Beispiel



- Syntaktische Regeln
 - Jedes DFD enthält mindestens eine Schnittstelle
 - Schnittstellen können aus Übersichtsgründen mehrfach gezeichnet werden
 - Keine Datenflüsse zwischen Schnittstellen
 - Jeder Datenfluss hat einen Namen
 - Ausnahmen: Datenflüsse von und zu Speichern transportieren die gesamten gespeicherten Daten
 - Keine direkten Datenflüsse zwischen Speichern
 - Keine direkten Datenflüsse zwischen Schnittstellen und Speichern

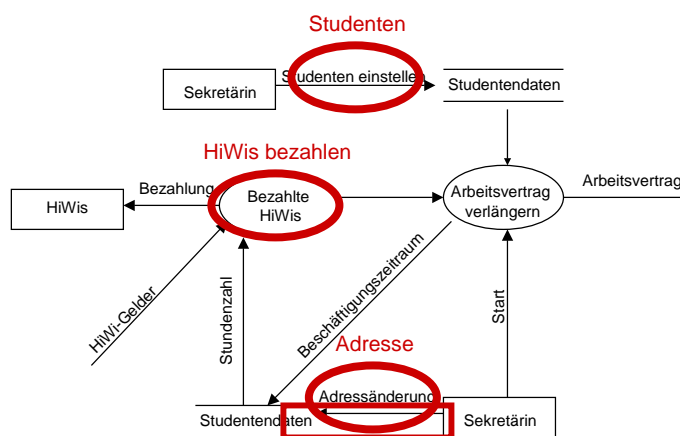
- Semantische Regeln
 - DFD beschreibt Datenfluss
 - *keine* Aussage über Kontrollfluss, d.h.
 - Terminierung
 - Initialisierung
 - Entscheidungen
 - Schleifen
 - Datenflussname als Substantiv (z.B. „neuer Versicherter“)
 - Funktionsname als Substantiv-Verb Kombination (z.B. „Anschrift erfassen“)
 - Funktionsnamen repräsentieren Aktionen
 - Die Annotationen der Kanten werden häufig benutzt, um die Struktur eines zugrundeliegenden Data Dictionaries abzuleiten.

9.1 Einführung

- Strukturierung
 - Funktionen können verfeinert werden, d.h. sie werden ihrerseits durch ein DFD beschrieben.
 - Datenflüsse können intern aufgespalten werden.
 - Auf diese Weise können DFDs auf mehreren Abstraktionsebenen dargestellt werden.

9.1 Einführung

- Welche Fehler sind in diesem DFD vorhanden?



Datenfluss zwischen Schnittstelle und Speicher

Aus: Balzert, H.: Lehrbuch der Software-Technik, Band 1, 2. Auflage, Spektrum, 2000

9.2 Fazit

- + DFDs leicht erstellbar und lesbar
- + hohe Intuitivität
- + DFD enthält mehr Informationen als Funktionsbaum
- DFDs zur Darstellung ganzer System zu unübersichtlich
- Einheitliches Abstraktionsniveau bei Daten und Funktionen schwierig einzuhalten
- Keine Darstellung des Datenaufbaus
- Keine formale Semantik
- Keine klaren Regeln für den Zustandsübergang

9.3 Literatur

- [DeM79] DeMarco, T.: Structured Analysis and System Specification, Yourdon Press, 1979