Höhere technische Lehranstalten
Abteilungen für Informatik

Datenbanken und Informationssysteme

Material for CLIL included

Kurt Hillebrand

September 2014

2 Konzeptionelles Datendesign

2.1 Grundbegriffe

- Konzeptionelles / Konzeptuelles Datenmodell / Datendesign, Entity-Relationship-Modelling (ERM), Datenstrukturanalyse (DSA), Logische / Semantische Datenmodellierung, Informationsmodell, Information Modelling

- Beschreibt einen Ausschnitt der realen Welt aus einer logischen Gesamtsicht

- Es wird ermittelt, welche Dinge (Entities) für eine Organisation / ein System bedeutend sind, welche Eigenschaften (Attribute) diese Dinge haben und wie sie zueinander in Beziehung (Relations) stehen.

- Modellierung mittels Constraints

- Vorteile:
  - stellt eine Gesamtschau auf die benötigte Information der Gesamtorganisation dar, geht daher über die lokale Sicht von Einzelanwendungen auf die Daten hinaus
  - ist eine gemeinsame sprachliche Basis für die Kommunikation zwischen den Anwendern und den Informatikern
  - ist unabhängig von der DV-technischen Implementierung, also auch vom verwendeten Datenbankmodell ('Struktogramm : Programm = Entity-Relationship-Diagramm : Daten')
  - bildet die Grundlage für die Ableitung der Datenstrukturen in ein konkretes Daten(bank)modell
  - Vergleich:

Konzeptionelles Datenmodell (Conceptual Data Model - CDM)	Physisches Datenmodell (Physical Data Model - PDM)
Darstellung unabhängig von einem bestimmten Datenbankmodell (z.B. Entity-Relationship-Modell)	Darstellung in einem bestimmten Datenbankmodell (z.B. Relationalem Datenbankmodell)
sollte der Anwender verstehen	irrelevant für den Anwender
- Entity-Typ - Attribut - Domäne (Domain) - Beziehungs-Typ	- Tabelle (Datei) - Spalte (Feld) - Datentyp - Fremdschlüssel - Index, Optimierung - Denormalisierung, Redundanzen

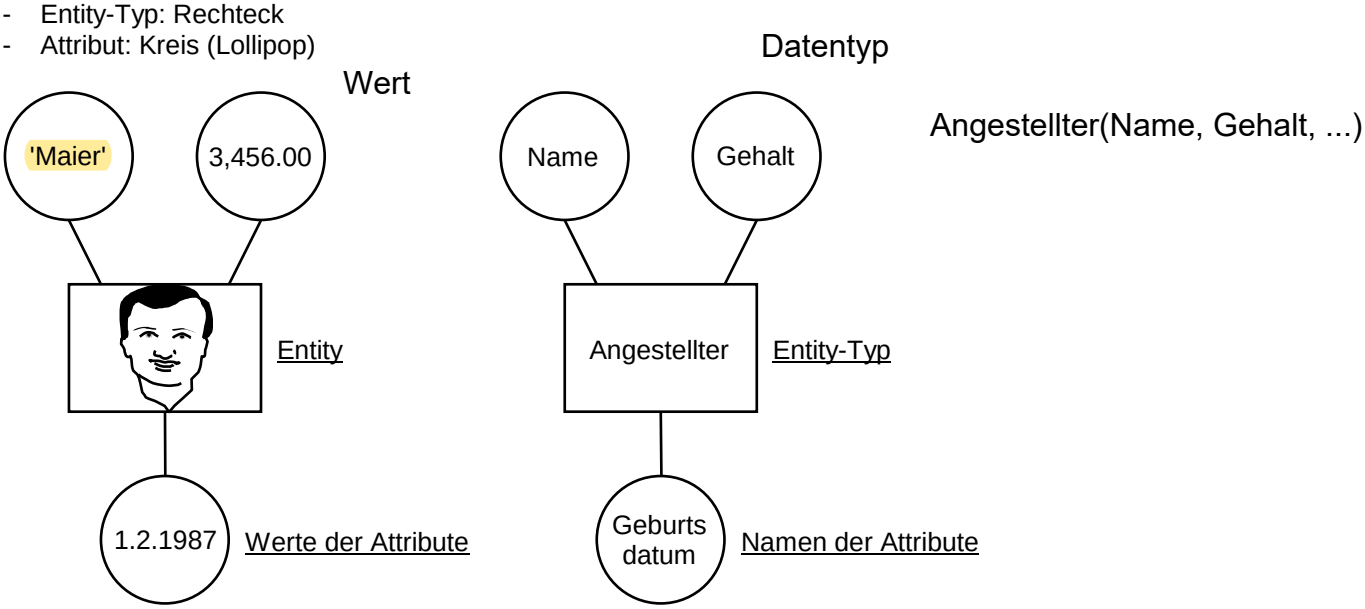
- Entity (Entität): Identifizierbare Einheit der realen Welt ('reale Wesenseinheit'), wie eine Person, ein Ding, ein Objekt, ein Ereignis, ein Begriff etc.
  Wesentlich ist, dass die verschiedenen Entities identifizierbar und damit voneinander unterscheidbar sind (Chen: A 'thing' which can be distinctly identified).
  Beispiele: ein Angestellter, ein Bauteil, ein Flug, eine Rechnungsverbuchung
  Gegenbeispiele: ein Schnee, eine Luft, eine Hoffnung

- Attribut (Eigenschaft): Merkmal eines Entities, das zu seiner Beschreibung wichtig ist.
  Jedes Attribut bekommt einen Namen.
  Beispiele: Name eines Angestellten, Gehalt eines Angestellten, Geburtsdatum eines Angestellten, Farbe eines Bauteils, Start- und Landezeitpunkt eines Fluges, Betrag einer Buchung

- Werte von Attributen: Für ein konkretes Entity hat jedes Attribut einen bestimmten Wert.
  Beispiele: für einen konkreten Angestellten ist der Name gleich 'Maier', ist der Gehalt gleich 3,456.00, ist das Geburtsdatum gleich 1.2.1987; für ein konkretes Bauteil ist die Farbe gleich 'rot'; für einen konkreten Flug ist der Startzeitpunkt gleich 24.12.2000 13:45; für eine konkrete Buchung ist der Betrag gleich 17.30

- Entity-Typ (Entity-Type, Entity-Menge, Entity-Set): Entities mit gleichen Attributen werden zu einem Entity-Typ zusammengefasst.
  Jeder Entity-Typ bekommt einen Namen, wobei üblicherweise die Einzahl verwendet wird (z.B. Angestellter, nicht Angestellte)
  Beispiele: Angestellte eines Unternehmens, in einem Unternehmen erzeugte Bauteile, Buchungen in einem Unternehmen

- Graphische Darstellung:
  - Entity-Typ: Rechteck
  - Attribut: Kreis (Lollipop)



- Das Entity selbst kann nicht dargestellt, sondern nur durch seine Attributwerte (eindeutig) beschrieben werden.

- Textliche Darstellung eines Entity-Typs mit seinen Attributen: (Attributschreibweise)
  Angestellter (Name, Gehalt, Geburtsdatum)

- Eindeutigkeit der Namen: In einem System müssen die Namen der Entity-Typen und der Attribute innerhalb der Entity-Typen eindeutig sein

- Auswertungen (d.h. Informationen, die aus den bestehenden Attributwerten abgeleitet werden können) sind nicht als Entities aufzufassen, z.B. Kundenliste

- Hinweise:
  - Die Begriffe Entity und Entity-Typ sowie Attribut und Attributwert sollen genau unterschieden werden.
  - Oft wird der Begriff Entity statt des Begriffs Entity-Typ verwendet, man spricht vom 'Entity Angestellter'.
  - Es ist zu unterscheiden, ob zwei Entities die gleichen Attribute oder die gleichen Attributwerte haben.
  - Beim Entity-Typ Angestellter ist das Attribut Geburtsdatum einem Attribut Alter vorzuziehen, da letzteres laufend geändert werden müsste.

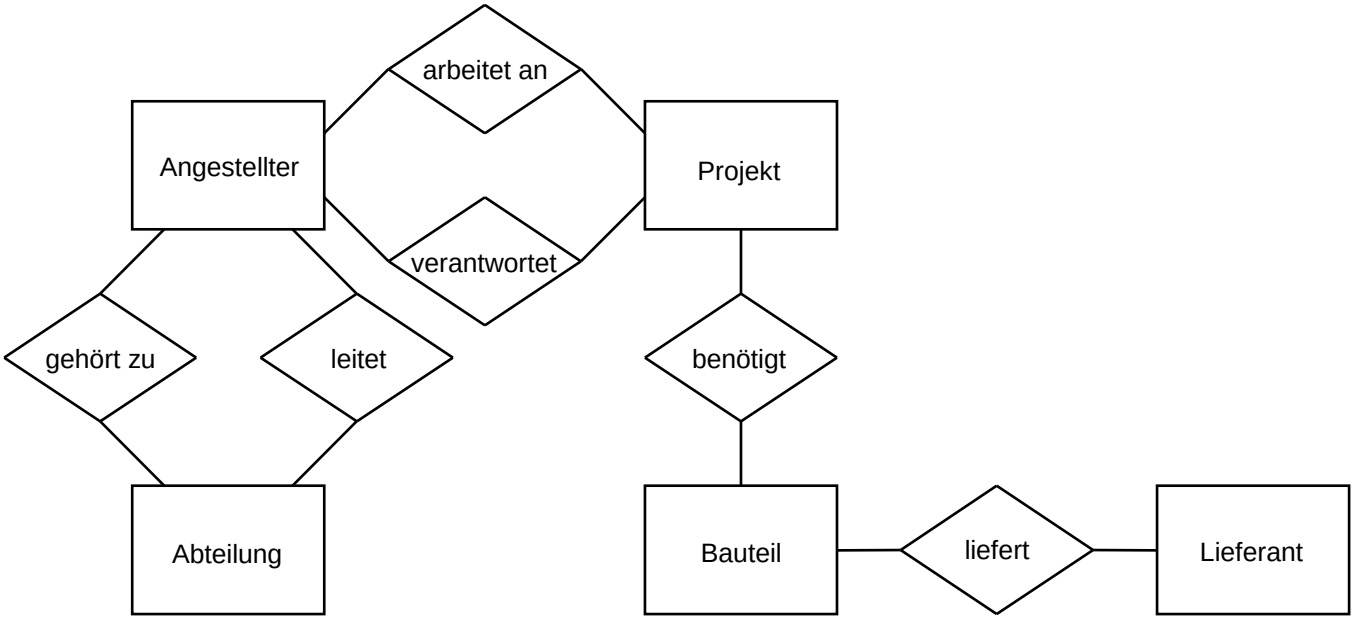
2.2 Entity-Relationship-Diagramm

- Beziehung (Relation, Relationship): Zwischen zwei (in der Folge auch mehr als zwei) Entities kann eine Beziehung (ein Zusammenhang) einer bestimmten Bedeutung bestehen.
  Beispiele: ein Angestellter konstruiert ein Bauteil, ein Angestellter tätigt eine Buchung

- Beziehungs-Typ (Relation-Type, Beziehungs-Menge, Relation-Set): Beziehungen, an denen Entities des gleichen Entity-Typs beteiligt sind und die dieselbe Bedeutung haben, werden zu einem Beziehungs-Typ zusammengefasst.
  Jeder Beziehungs-Typ bekommt einen Namen (meist ein Zeitwort).
  Beispiele: bestimmte Angestellte konstruieren bestimmte Bauteile, bestimmte Buchungen werden von bestimmten Angestellten getätigt

- Der Begriff Relationship ist ein Überbegriff für Relationship-Instance (Beziehung) und Relationship-Type (Beziehungs-Typ)

- Entity-Relationship-Diagramm, ER-Diagramm (nach P. Chen, 1976):
  - Entity-Typ: Rechteck
  - Beziehungs-Typ: Raute mit Linien zu den, am Beziehungs-Typ beteiligten, Entity-Typen
  - Attribute von Entity-Typen werden in ER-Diagramme, wegen der Übersichtlichkeit, üblicherweise nicht eingezeichnet (sondern in textlicher Darstellung als Legende angegeben)



- Hinweis: Bauteil ist bei diesem Beispiel nicht als physisch konkretes Einzelbauteil, sondern als Bauteil-Art zu verstehen. Es wird nicht jede einzelne M5 Schraube verwaltet, sondern die Schraubenart M5 (ev. mit Anzahl als Attribut)

- Ein Entity-Typ kann an mehreren Beziehungs-Typen beteiligt sein.
  Beispiel: Entity-Typ 'Projekt', Beziehungs-Typen 'verantwortet' und 'benötigt'

- Zwischen zwei Entity-Typen können mehrere Beziehungs-Typen (verschiedener Bedeutung) bestehen (Parallelstrukturen).
  Beispiel: Entity-Typen 'Angestellter' und 'Abteilung', Beziehungs-Typen 'gehört zu' und 'leitet'

- Benötigte Informationen, die aus bestehenden Beziehungen abgeleitet werden können, dürfen nicht als eigene Beziehung definiert werden (Überbestimmtheit).
  Beispiel: Die Information 'Welche Abteilungen wirken bei welchen Projekten mit?' kann aus den Beziehungs-Typen 'gehört zu' und 'arbeitet an' abgeleitet werden, es ist kein eigener Beziehungs-Typ notwendig.

- Die Existenz eines (graphentheoretischen) Weges zwischen zwei Entity-Typen ist jedoch keine Gewähr, dass Beziehungsinformationen zwischen diesen beiden Entity-Typen abgeleitet werden können.
  Beispiel: Die Information 'Welche Lieferanten beliefern welche Projekte?' kann aus den Beziehungs-Typen 'benötigt' und 'liefert' nicht abgeleitet werden

- Hinweise:
  - Die Begriffe Beziehung und Beziehungs-Typ sollen genau unterschieden werden.
  - Oft wird der Begriff Beziehung statt des Begriffs Beziehungs-Typ verwendet, man spricht von der 'Beziehung leitet'.

- Implementierungs- und Performanceüberlegungen sind in diesem Stadium der Datenmodellierung nicht anzustellen

2.3 Komplexitätsgrad von Beziehungs-Typen

2.3.1 Drei Arten von Beziehungs-Typen

- Die Art des Beziehungs-Typs wird auch Beziehungskardinalität (Cardinality), Konnektivität oder Komplexität eines Beziehungs-Typs genannt.

- 1:1-Beziehungs-Typ (eins zu eins-Beziehungs-Typ): Jedem Entity des einen Entity-Typs ist höchstens ein Entity des anderen Entity-Typs zugeordnet und umgekehrt.

  - Beispiele:

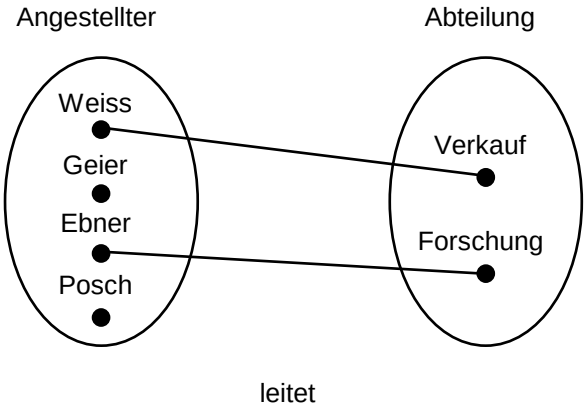
```
[Angestellter] ---1--- <leitet> ---1--- [Abteilung]
[Mann] ---1--- <ist verheiratet mit> ---1--- [Frau]
[Person] ---1--- <besitzt> ---1--- [Österreichischer Führerschein]
[Mitarbeiter] ---1--- <hat> ---1--- [Firmenauto]
```

- Ein Beziehungs-Typ R zwischen den beiden Entity-Typen E1 und E2 kann als Mathematische Relation (Menge von Paaren oder Tupeln)
  $R \subseteq E1 \times E2$
  aufgefasst werden, mit
  $E1 \times E2 = \{ (e1,e2) / e1 \in E1 \& e2 \in E2 \}$, wobei gilt
  (e1,e2) ist genau dann ein Element von R, wenn e1 und e2 in Beziehung stehen.

- In einem 1:1-Beziehungs-Typ muss die mathematische Relation eine Funktion mit existierender Umkehrfunktion sein.

- Die Entities der beiden Entity-Typen kommen in den Paaren der Relation höchstens einmal vor.

- Beispiel (Mengendiagramm und Mengenschreibweise):



  R = { (Weiss,Verkauf), (Ebner,Forschung) }

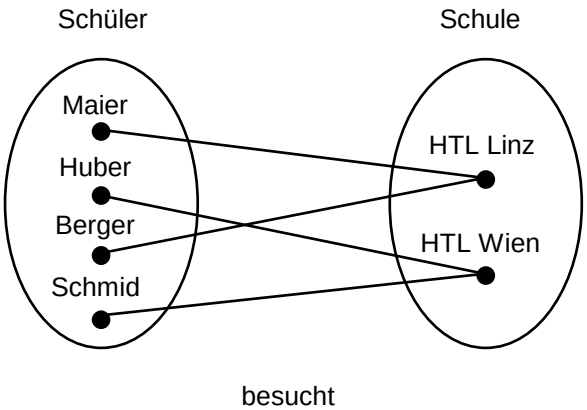
- 1:n-Beziehungs-Typ (eins zu viele-Beziehungs-Typ, hierarchischer Beziehungs-Typ): Jedem Entity des einen Entity-Typs ist höchstens ein Entitiy des anderen Entity-Typs zugeordnet, nicht aber umgekehrt.

  - Beispiele:
    ```
    [Bezirk] ---1--- <bewohnt von> ---n--- [Niederösterreicher]
    [Schüler] ---n--- <besucht> ---1--- [Schule]
    ```

  - In einer 1:n-Beziehung muss die mathematische Relation eine Funktion sein.

  - Die Entities des einen Entity-Typs kommen in den Paaren der Relation höchstens einmal vor, die des anderen Entity-Typs kommen mehrmals vor.

  - Beispiel (Mengendiagramm und Mengenschreibweise):



  R = { (Maier,HTL Linz), (Huber,HTL Wien), (Berger,HTL Linz), (Schmid,HTL Wien) }

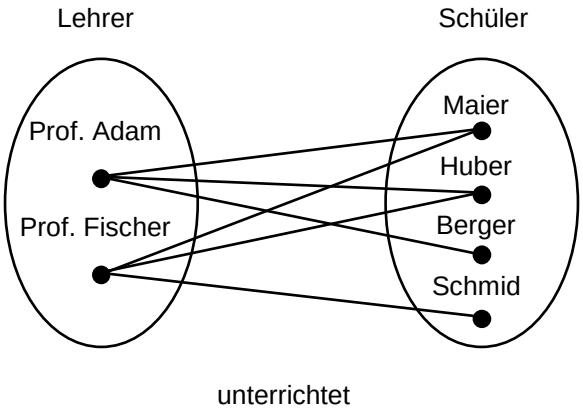
- m:n-Beziehungs-Typ (viele zu viele-Beziehungs-Typ, netzwerkartiger Beziehungs-Typ): Keine Einschränkungen in der Zuordnung zwischen den Entities der beiden Entity-Typen.

  - Beipiele:
    ```
    [Lehrer] ---m--- <unterrichtet> ---n--- [Schüler]
    [Bauteil] ---m--- <wird geliefert von> ---n--- [Lieferant]
    ```

  - In einer m:n-Beziehung muss die mathematische Relation keinen Bedingungen genügen.

  - Die Entities der beiden Entity-Typen kommen in den Paaren der Relation mehrmals vor.

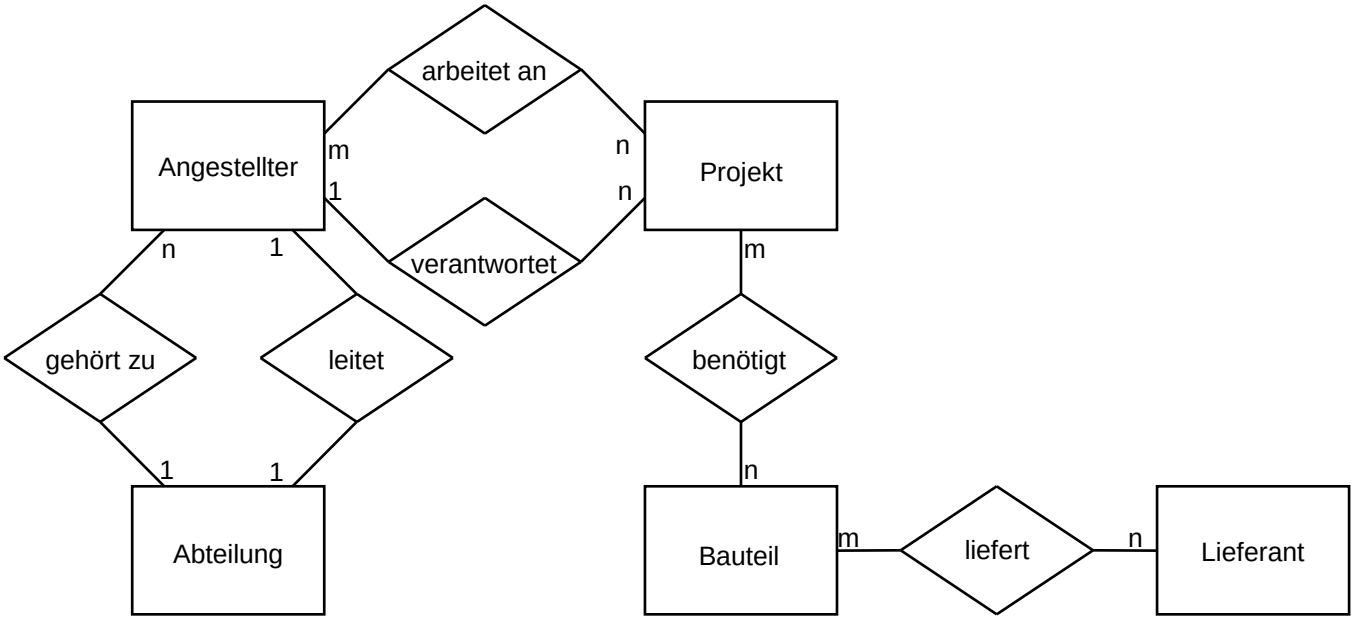
  - Beispiel (Mengendiagramm und Mengenschreibweise):



R = { (Prof. Adam,Maier), (Prof. Adam,Huber), (Prof. Adam,Berger),
(Prof. Fischer,Maier), (Prof. Fischer,Huber), (Prof. Fischer,Schmid) }

2.3.2 ER-Diagramm mit Arten der Beziehungs-Typen

- Beachte: 1:n-Beziehung ist nicht symmetrisch, es ist wesentlich wo 1 und wo n steht!



- Übung: Wie ändern sich die Komplexitätsgrade, wenn Bauteil nicht im Sinne von Bauteil-Art, sondern als Einzelbauteil aufgefasst wird?

- Übung: Wie ändern sich die Komplexitätsgrade, wenn es zu jedem Bauteil nur einen (Stamm-)Lieferant gibt?

2.3.3 (min,max)-Notation

- Charakterisiert den Beziehungs-Typ über die Art hinausgehend noch genauer.

- Jedem Entity-Typ wird für jedem Beziehungs-Typ, an dem er beteiligt ist, ein Zahlenpaar (min,max), mit folgender Bedeutung zugeordnet:
  Jedes Entity des Entity-Typs ist mindestens an min und höchstens an max Beziehungen des Beziehungs-Typs beteiligt. Offensichtlich muss gelten: $0 \leq min \leq max \ \& \ max \geq 1$
- Ein beliebig großes max wird mit n symbolisiert.

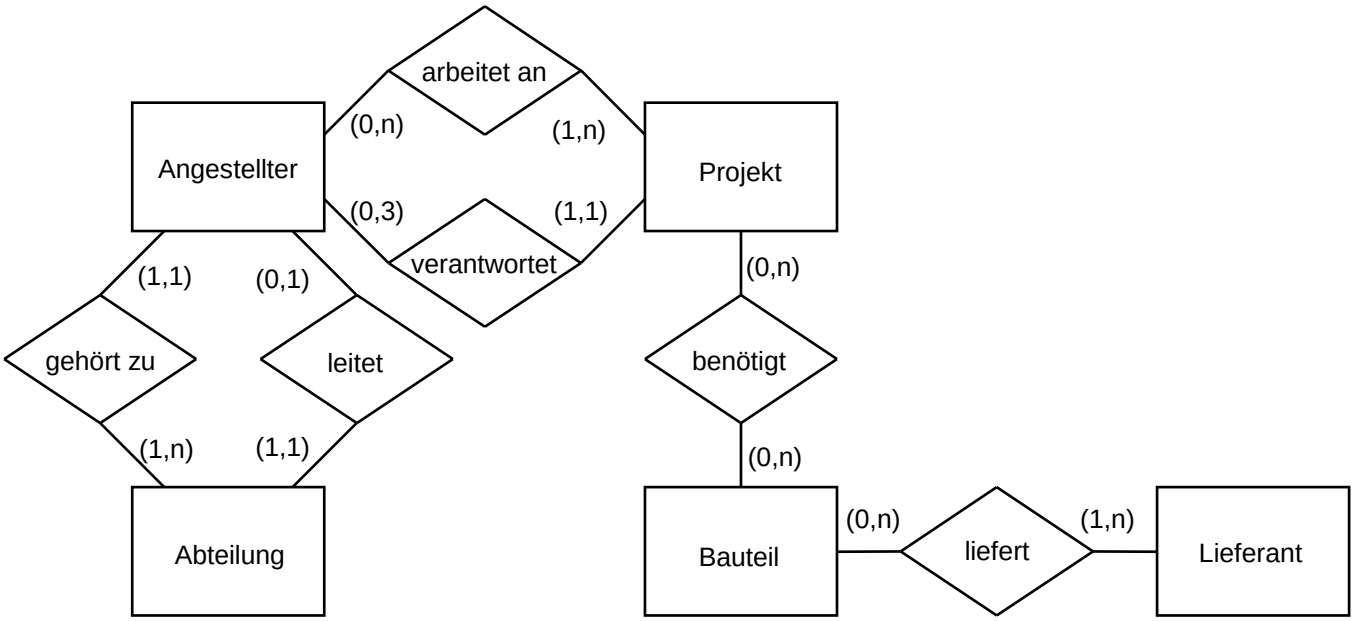
- Beziehungs-Typen, die als min=0 enthalten, werden auch Konditionelle Beziehungs-Typen (Kann-Beziehungs-Typen, Partial Relation-Type) genannt.

- Ein Entity-Typ, für den bei einem Beziehungs-Typ min=0 definiert ist, nimmt an diesem Beziehungs-Typ teilweise teil (Partial oder Optional Participation).

- Ein Entity-Typ, für den bei einem Beziehungs-Typ min>0 definiert ist, nimmt an diesem Beziehungs-Typ vollständig teil (Total oder Mandatory Participation).

- Bei einigen Notationsformen sind nur die Werte 0,1 und n für min und max erlaubt.

- Beispiele:
```
[Angestellter] ---(0,1)--- <leitet> ---(1,1)--- [Abteilung]
[Mann] ---(0,1)--- <ist verheiratet mit> ---(0,1)--- [Frau]
[Person] ---(0,1)--- <besitzt> ---(1,1)--- [Österreichischer Führerschein]
[Mitarbeiter] ---(0,1)--- <hat> ---(1,1)--- [Firmenauto]
[Bezirk] ---(1,n)--- <bewohnt von> ---(1,1)--- [Niederösterreicher]
[Schüler] ---(0,1)--- <besucht> ---(1,n)--- [Schule]
[Lehrer] ---(1,n)--- <unterrichtet> ---(1,n)--- [Schüler]
[Bauteil] ---(0,n)--- <wird geliefert von> ---(1,n)--- [Lieferant]
```
  (es gibt auch eigengefertigte Bauteile)



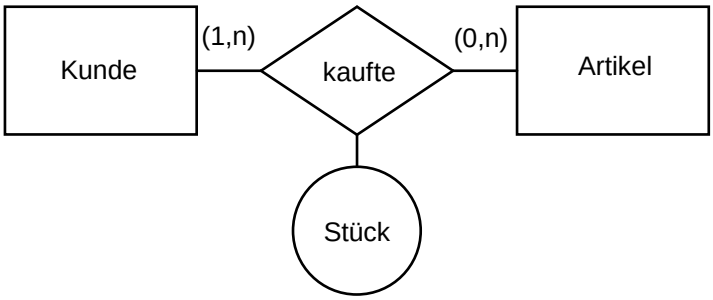
2.4 Attribute von Beziehungen

- Beziehungs-Typen können auch Attribute als Eigenschaften haben. Für eine Beziehung nimmt jedes Attribut einen bestimmten Wert an.

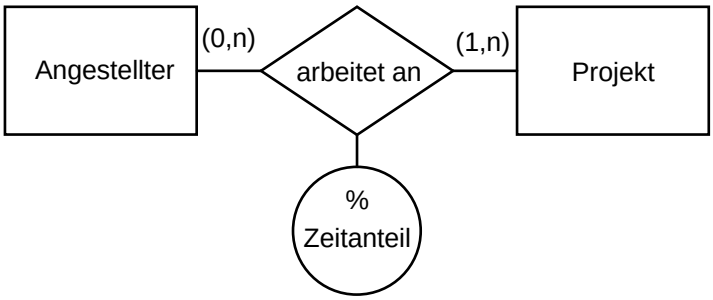
- Attribute sind nur bei m:n-Beziehungs-Typen sinnvoll. Andernfalls kann das Attribut beim Entity-Typ auf der Seite mit max=1 ist dazugegeben werden.

- Attribute von Beziehungs-Typen werden in ER-Diagramme, wegen ihrer Besonderheit, üblicherweise schon eingezeichnet.

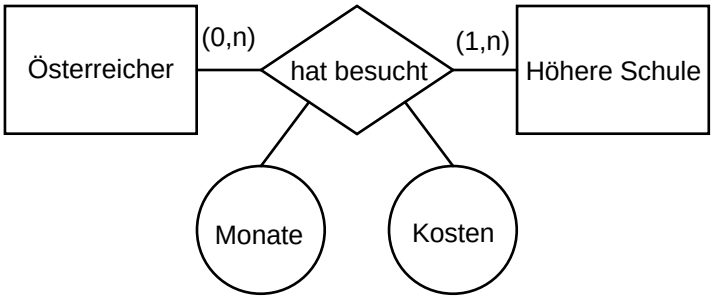
- Beispiel: Kunde hat soviel Stück dieses Artikels gekauft



- Beispiel: Angestellter arbeitet einen bestimmten Zeitanteil bei diesem Projekt mit (Randbedingung: Summen der Zeitanteile eines Angestellten ≤ 100)



- Beispiel: Österreicher besucht die Höhere Schule eine bestimmte Anzahl von Monaten und verursacht dabei bestimmte Kosten



2.5 Beziehungs-Typ oder Entity-Typ

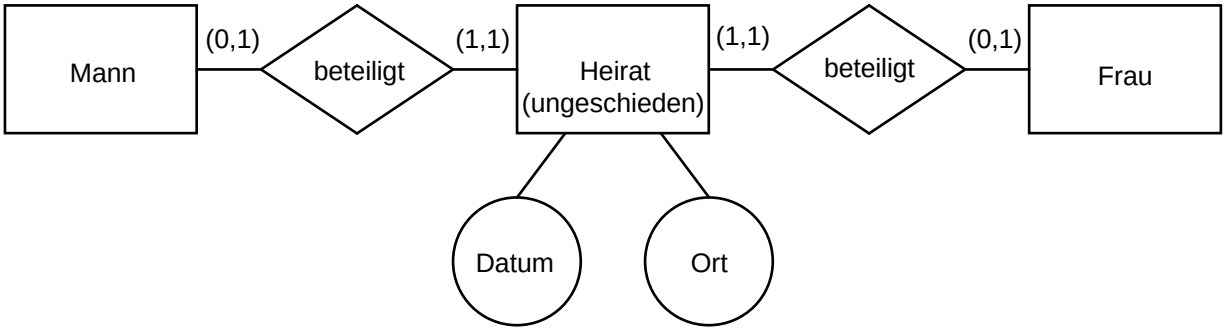
- Ein Sachverhalt, der durch eine Beziehung beschrieben ist, kann meist auch als Entity aufgefasst werden. Dies trifft besonders auf Beziehungen mit Attributen zu.

- Welche Variante zu wählen ist ergibt sich logisch aus der Aufgabenstellung, muss aber teilweise auch willkürlich festgelegt werden.

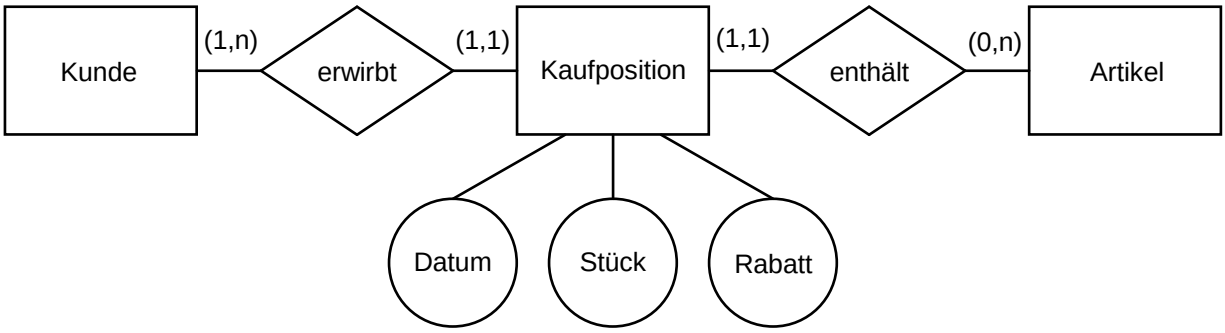
- Die Beziehung wird meist zum Ereignis (und damit Entity), durch das die Beziehung zustandegekommen ist, umfunktioniert.

- Die verbleibenden Beziehungs-Typen haben keine Attribute mehr.

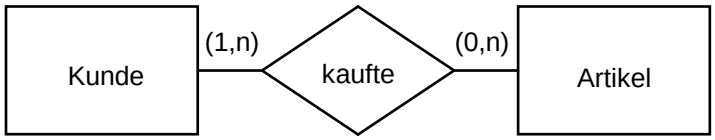
- Beispiel: Der Mann hat die Frau an einem bestimmten Datum, in einem bestimmten Ort geheiratet

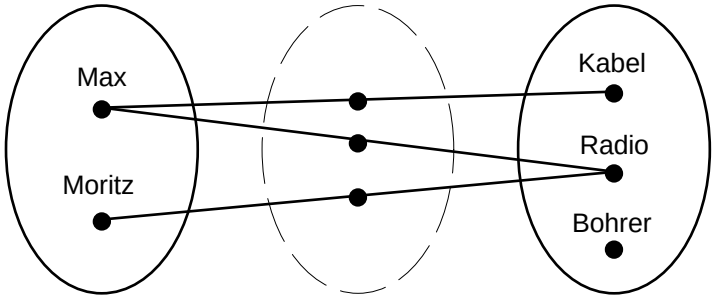


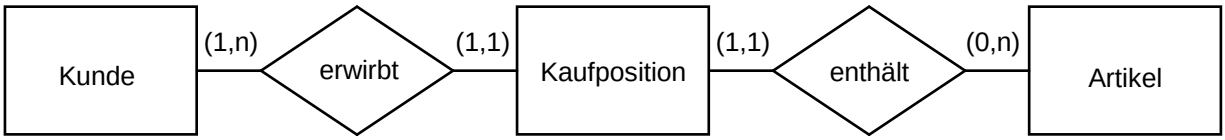
- Beispiel: Der Kunde hat an einem bestimmten Datum soviel Stück des Artikels mit diesem Rabatt gekauft



- Auf diese Art sind auch m:n-Beziehungs-Typen in einen Entity-Typ und zwei 1:n-Beziehungs-Typen auflösbar:







- Für den zusätzlichen Entity-Typ und für die beiden 1:n-Beziehungs-Typen müssen sinnvolle Bezeichnungen gefunden werden.

- Ein so entstandener Entity-Typ wird auch als Assoziativer Entity-Typ (Associative Entity-Type, Intersection Entity-Type) bezeichnet.

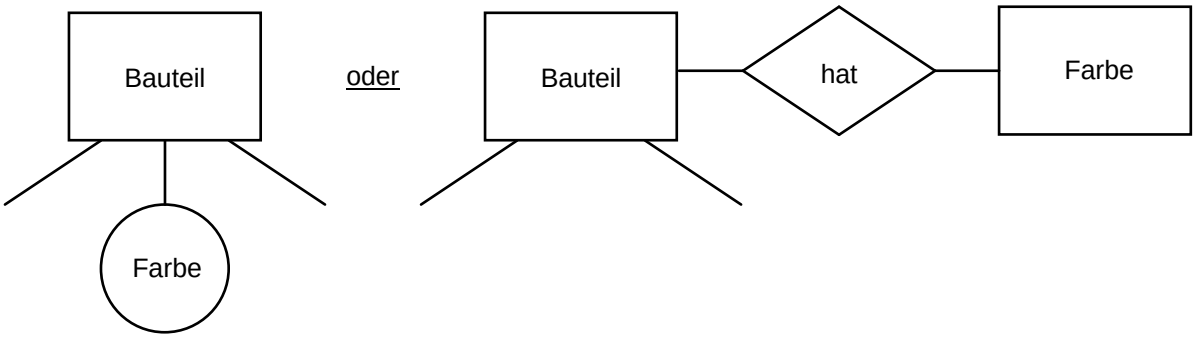
- Die (min,max)-Werte an den ursprünglichen Entity-Typen bleiben gleich, die zwei neuen sind immer (1,1)

2.6 Attribut oder Entity-Typ

- Es gibt Aufgabenstellungen, in denen eine Objekteigenschaft als Attribut oder als eigener Entity-Typ definiert werden kann.

- Wenn eine Eigenschaft eines Entity-Typs auch als eigenständige, nicht nur an das Objekt gebundene, Größe zu behandelt ist, so ist diese als eigener Entity-Typ in das Datenmodell aufzunehmen.

- Beispiel:

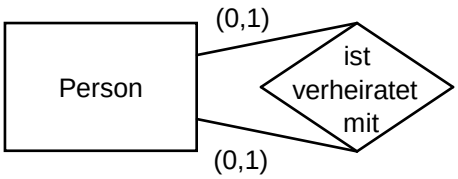


  - Wenn eine der folgenden Bedingungen zutrifft, sollte Farbe als eigener Entity-Typ modelliert werden:
    - Bauteile haben mehrere Farben
    - Farben werden noch zusätzlich beschrieben (haben Eigenschaften)
    - die möglichen Werte (des Attributes) sollen auf bestimmte Begriffe eingeschränkt werden (z.B. Schlagworte, Wissensgebiete)

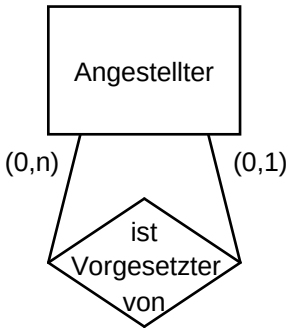
2.7 Beziehungen zwischen Entities desselben Typs

- Beziehungen sind auch zwischen Enities desselben Typs möglich. Man spricht dann auch von Rekursiven oder Reflexiven Beziehungen.

- Beispiel:

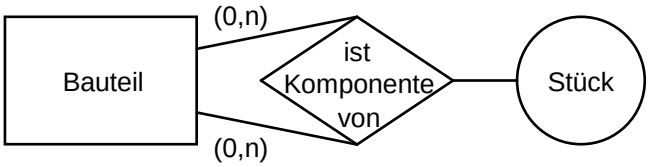


- Beispiel:



  - 0 in (0,n) weil nicht jeder Angestellte Vorgesetzter ist ('Die Hackler')
  - 0 in (0,1) weil es zumindest einen Angestellten geben wird, der keinen Vorgesetzten hat ('Der Big Boss')

- Beispiel:



  - Stücklistenproblematik
  - Konkretes Beispiel dazu:
    Bauteile: B1=Fahrrad Type 1, B2= Fahrrad Type 2, B3=Rahmen, B4=Lenker, B5=Rad, B6=Mantel, B7=Schlauch, B8= Felge, B9=Speiche, B10=Nabe,...
    ist Komponente von = { (B3,B1,1), (B4,B1,1), (B4,B2,1), (B5,B1,2), (B6,B5,1), (B7,B5,1), (B8,B5,1), (B9,B5,24), (B10,B5,1), ...}
  - 0 in (0,n) oben: Bauteil kommt in keinem anderen Bauteil vor ('Fahrrad Type 1')
  - n in (0,n) oben: Bauteil kommt in mehreren anderen Bauteilen vor ('Lenker')
  - 0 in (0,n) unten: Bauteil besteht aus keinen weiteren Bauteilen mehr ('Speiche')
  - n in (0,n) unten: Bauteil besteht aus mehreren anderen Bauteilen ('Rad')

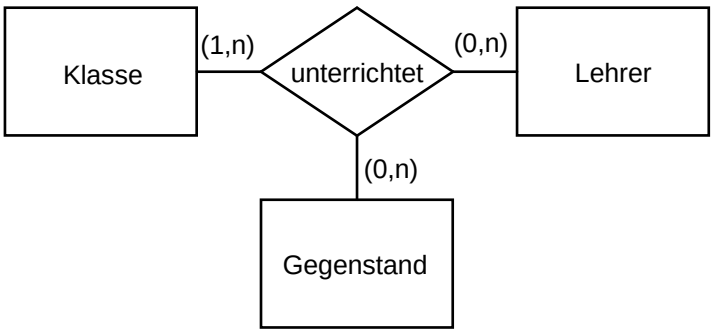
- Leseart: von links nach rechts oder von oben nach unten

2.8 Beziehungen zwischen mehr als zwei Entity-Typen

- Die Anzahl der Entities, die an einer Beziehung beteiligt sind, wird Grad oder Wertigkeit der Beziehung (Degree) genannt.
  Bis jetzt wurden nur Beziehungs-Typen mit Grad=2 (Zweiwertige oder Binäre Beziehungs-Typen) besprochen, es treten aber auch Beziehungs-Typen mit höheren Graden (Mehrwertige Beziehungs-Typen) häufig auf.

- Wenn der Grad des Beziehungs-Typs gleich g ist so besteht die Relation, die den Beziehungs-Typ darstellt, aus Tupeln mit g Elementen (bei Binären Beziehungs-Typen aus Paaren).

- Beispiel:



  unterrichet = { (3A,Eva,D), (3A,Eva,E), (3A,Max,E), (3B,Eva,D), (3B,Eva,E), (3B,Max,E),... }

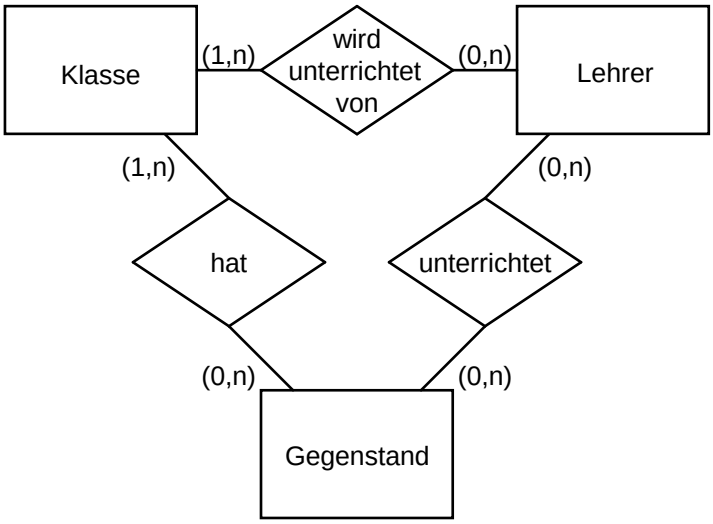
- Übung: Geben Sie plausible Erklärungen für die min-Werte in obigem Beispiel an

- Natürlich sind auch in diesem Fall Attribute beim Beziehungs-Typ möglich, Beispiel: Stundenanzahl

- max-Werte müssen alle n sein, sonst ist ein mehrwertiger Beziehungs-Typ nicht notwendig

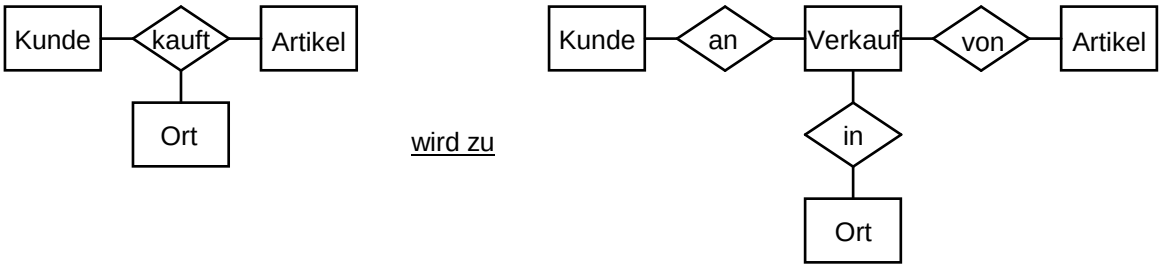
- Übung: Erstellen Sie ERDs mit Angabe der (min,max)-Werte für die folgenden Situationen
  Projekt, Bauteil, Lieferant: Lieferant liefert Bauteil für Projekt
  Angestellter, Angestellter, Projekt: Beim Projekt ist der Angestellte dem Angestellten unterstellt

- Es ist zu beachten, dass obiges Beispiel, definiert mit drei Binären Beziehungs-Typen, einen anderen Informationsgehalt hat:



wird unterrichtet von = {(3A,Eva), (3A,Max), (3B,Eva), (3B,Max),... }
hat = {(3A,D), (3A,E), (3B,D), (3B,D),... }
unterrichtet = {(Eva,D), (Eva,E), (Max,E),... }

- Ein Mehrwertiger Beziehungs-Typ kann eliminiert und durch binäre Beziehungs-Typen ersetzt werden:



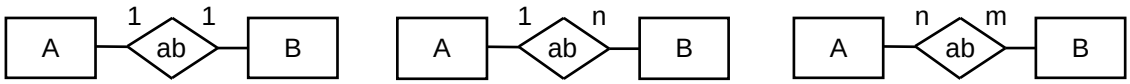
  - Die (min,max)-Werte an den ursprünglichen Entity-Typen bleiben gleich, die neuen sind immer (1,1)

2.9 Weitere Notationsformen von ER-Diagrammen

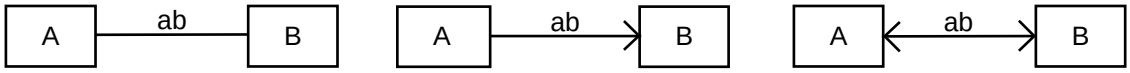
- Es gibt eine Vielzahl von Notationsformen für ER-Diagramme

- Außer den bisher verwendeten Chen-Diagrammen sind die Diagramme nach C. Bachman weit verbreitet. Bei diesen wird ein Beziehungs-Typ durch eine Linie dargestellt, an der der Name des Beziehungs-Typs vermerkt ist. Damit sind folgende Einschränkungen gegeben:
  - Beziehungs-Typ mit Attributen ist nicht möglich, ein solcher muss als Entity-Typ dargestellt werden.
  - Es sind nur Binäre Beziehungs-Typen möglich, solche mit höherem Grad müssen als Entity-Typen definiert werden.

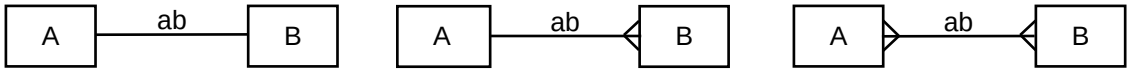
Chen-Diagramm



Bachman-Diagramm mit Arrows

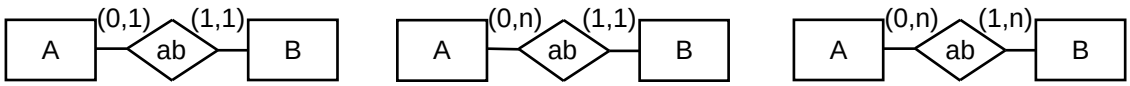


Bachman-Diagramm mit Crowfeet

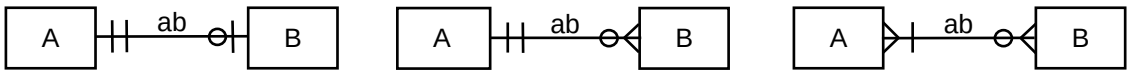


- In einem Diagramm entweder Arrows oder Crowfeet verwenden (nicht mischen)
- Ein rekursiver Beziehungs-Typ mit Crowfeet wird auch 'Pig Ear' genannt
- Die (min,max)-Notation wird auch mit Symbolen (und damit eingeschränkt) verwendet:
  - Die Symbole sind auf der anderen Seite der Linie als die (min,max)-Angaben
  - Das Symbol für max ist immer am Rechteck
  - Symbol für 0: Kreis
  - Symbol für 1: Linie
  - Symbol für n: Pfeilspitze oder umgekehrte Pfeilspitze

Chen-(min,max)-Notation (same-side cardinality)



Bachman-Symbol-Notation (look-across cardinality)



- In weiteren Varianten werden die Attribute im Rechteck notiert
  (Name des Entity-Typs, auch fett, wird mit einer Linie von den Namen der Attribute getrennt)
  - Vergleiche: Klassendiagramme (Class Diagrams) in UML (Unified Modelling Language)
  - Vorteile: Übersichtlichkeit, weniger Graphikelemente
  - Nachteile: Rechtecke werden unterschiedlich groß, bei Attributen von Beziehungs-Typen nicht verwendbar
  - Assoziation: Bezeichnung für den Beziehungs-Typ
  - Multiplizität: Bezeichnung für die (min,max)-Werte

  - Beispiel: