

Teil III

Entity-Relationship-Modell

# Entity-Relationship-Modell

- 1 Datenbankmodell
- 2 ER-Modell
- 3 Weitere Konzepte im ER-Modell

## Lernziele für heute ...

- Kenntnis der Konzepte des Entity-Relationship-Modells
- Fähigkeiten zur konzeptuellen Modellierung eines Anwendungsbereichs



# Grundlagen von Datenbankmodellen

Ein **Datenbankmodell** ist ein System von Konzepten zur Beschreibung von Datenbanken. Es legt Syntax und Semantik von Datenbankbeschreibungen für ein Datenbanksystem fest.

- Datenbankbeschreibungen = Datenbankschemata

# Ein Datenbankmodell legt fest...

## 1 statische Eigenschaften

- 1 Objekte
- 2 Beziehungen

inklusive der Standard-Datentypen, die Daten über die Beziehungen und Objekte darstellen können,

## 2 dynamische Eigenschaften wie

- 1 Operationen
- 2 Beziehungen zwischen Operationen,

## 3 Integritätsbedingungen an

- 1 Objekte
- 2 Operationen

# Datenbankmodelle

- Klassische Datenbankmodelle sind speziell geeignet für
  - ▶ große Informationsmengen mit relativ starrer Struktur und
  - ▶ die Darstellung statischer Eigenschaften und Integritätsbedingungen (also die Bereiche 1(a), 1(b) und 3(a))
- Entwurfsmodelle: (E)ER-Modell, UML, ...
- Realisierungsmodelle: Relationenmodell, objektorientierte Modelle, ...

# Datenbanken versus Programmiersprachen

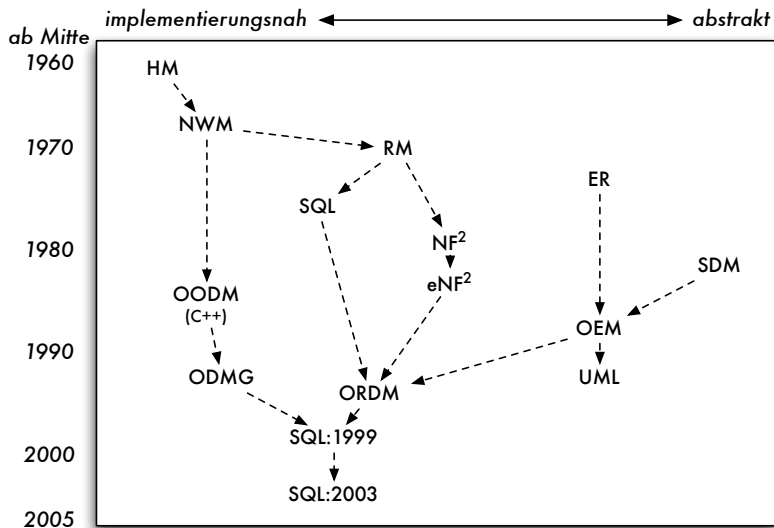
<b>Datenbankkonzept</b>	<b>Typsystem einer Programmiersprache</b>
Datenbankmodell <i>Relation, Attribut ...</i>	Typsystem <b>int, struct ...</b>
Datenbankschema <b>relation</b> WEIN = (...)	Variablendeklaration <b>var</b> x: <b>int</b> , y: <b>struct</b> Wein
Datenbank WEIN(4961, 'Chardonnay', 'Weiß', ...)	Werte 42, 'Cabernet Sauvignon'

# Abstraktionsstufen

<b>Modelle</b>	<b>Daten</b>	<b>Algorithmen</b>
abstrakt	Entity-Relationship-Modell	Struktogramme
konkret	Hierarchisches Modell Netzwerkmodell Relationenmodell	Pascal C, C++ Java, C#



# Datenbankmodelle im Überblick



# Datenbankmodelle im Überblick /2

- HM: hierarchisches Modell, NWM: Netzwerkmodell, RM: Relationenmodell
- $NF^2$ : Modell der geschachtelten (Non-First-Normal-Form =  $NF^2$ ) Relationen,  $eNF^2$ : erweitertes  $NF^2$ -Modell
- ER: Entity-Relationship-Modell, SDM: semantische Datenmodelle
- OODM / C++: objektorientierte Datenmodelle auf Basis objektorientierter Programmiersprachen wie C++, OEM: objektorientierte Entwurfsmodelle (etwa UML), ORDM: objektrelationale Datenmodelle

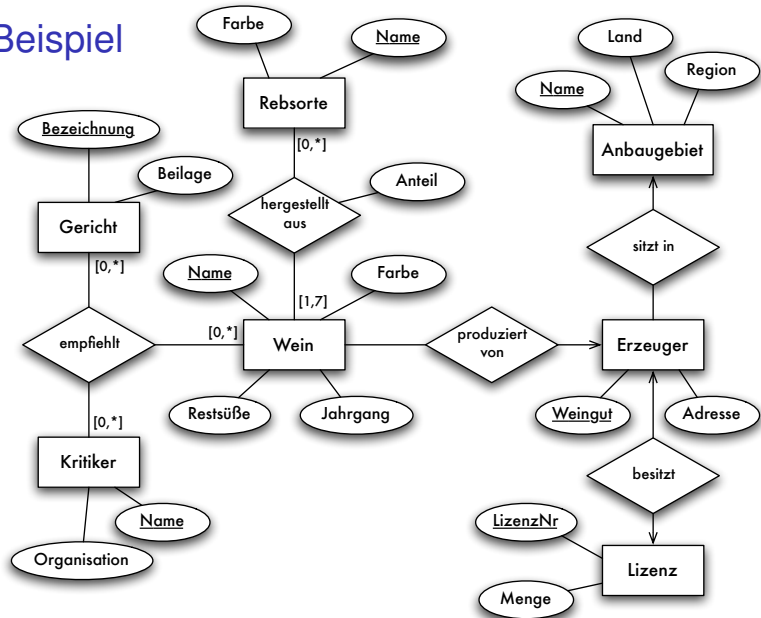
# Das ER-Modell

**Entity:** Objekt der realen oder der Vorstellungswelt, über das Informationen zu speichern sind, z.B. **Produkte** (Wein, Katalog), Winzer oder Kritiker; aber auch Informationen über Ereignisse, wie z.B. **Bestellungen**

**Relationship:** beschreibt eine Beziehung zwischen Entities, z.B. ein Kunde **bestellt** einen Wein oder ein Wein wird von einem Winzer **angeboten**

**Attribut:** repräsentiert eine Eigenschaft von Entities oder Beziehungen, z.B. **Name** eines Kunden, **Farbe** eines Weines oder **Datum** einer Bestellung

## ER-Beispiel



# Werte

- **Werte**: primitive Datenelemente, die direkt darstellbar sind
- Wertemengen sind beschrieben durch **Datentypen**, die neben einer Wertemenge auch die Grundoperationen auf diesen Werten charakterisieren
- ER-Modell: vorgegebene Standard-Datentypen, etwa die ganzen Zahlen **int**, die Zeichenketten **string**, Datumswerte **date** etc.
- jeder Datentyp stellt Wertebereich mit Operationen und Prädikaten dar

# Entities

- **Entities** sind die in einer Datenbank zu repräsentierenden Informationseinheiten
- im Gegensatz zu Werten nicht direkt darstellbar, sondern nur über ihre Eigenschaften beobachtbar
- Entities sind eingeteilt in **Entity-Typen**, etwa  $E_1, E_2 \dots$

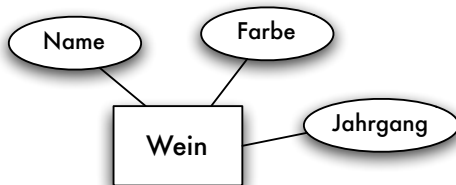
A rectangular box with a black border and a white background, containing the word "Wein" in black text. The box is centered on the slide.

Wein

- Menge der aktuellen Entities:  $\mathcal{E} = \{e_1, e_2, \dots, e_n\}$

# Attribute

- **Attribute** modellieren Eigenschaften von Entities oder auch Beziehungen
- alle Entities eines Entity-Typs haben dieselben Arten von Eigenschaften; Attribute werden somit für Entity-Typen deklariert



- textuelle Notation  $E(A_1 : D_1, \dots, A_m : D_m)$

# Identifizierung durch Schlüssel

- Schlüsselattribute: Teilmenge der gesamten Attribute eines Entity-Typs  $E(A_1, \dots, A_m)$

$$\{S_1, \dots, S_k\} \subseteq \{A_1, \dots, A_m\}$$

- in jedem Datenbankzustand identifizieren die aktuellen Werte der Schlüsselattribute eindeutig Instanzen des Entity-Typs  $E$
- bei mehreren möglichen Schlüsselkandidaten: Auswahl eines **Primärschlüssels**
- Notation: markieren durch Unterstreichung:

$$E(\dots, \underline{S_1}, \dots, \underline{S_i}, \dots)$$



# Beziehungstypen

- Beziehungen zwischen Entities werden zu **Beziehungstypen** zusammengefasst
- allgemein: beliebige Anzahl  $n \geq 2$  von Entity-Typen kann an einem Beziehungstyp teilhaben
- zu jedem  $n$ -stelligen Beziehungstyp  $R$  gehören  $n$  Entity-Typen  $E_1, \dots, E_n$
- Ausprägung  $\mathcal{R}$  eines Beziehungstyps

$$\mathcal{R} \subseteq \mathcal{E}_1 \times \mathcal{E}_2 \times \dots \times \mathcal{E}_n$$

## Beziehungstypen /2

- Notation

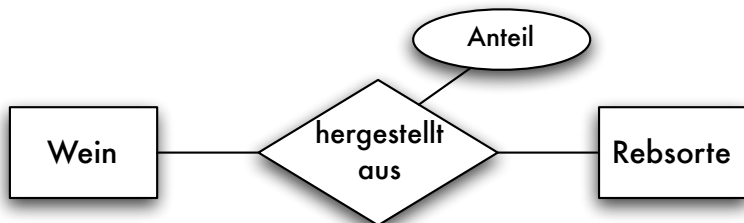


- textuelle Notation:  $R(E_1, E_2, \dots, E_n)$
- wenn Entity-Typ mehrfach an einem Beziehungstyp beteiligt:  
Vergabe von **Rollennamen** möglich

verheiratet(Frau: Person, Mann: Person)

# Beziehungsattribute

- Beziehungen können ebenfalls Attribute besitzen
- Attributdeklarationen werden beim Beziehungstyp vorgenommen; gilt auch hier für alle Ausprägungen eines Beziehungstyps  $\rightsquigarrow$   
**Beziehungsattribute**

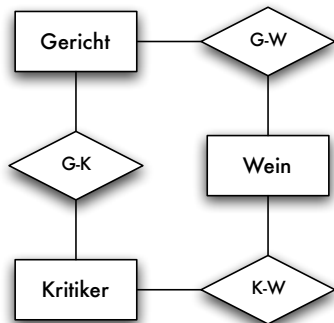
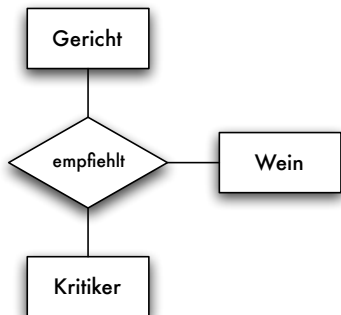


- textuelle Notation:  $R(E_1, \dots, E_n; A_1, \dots, A_k)$

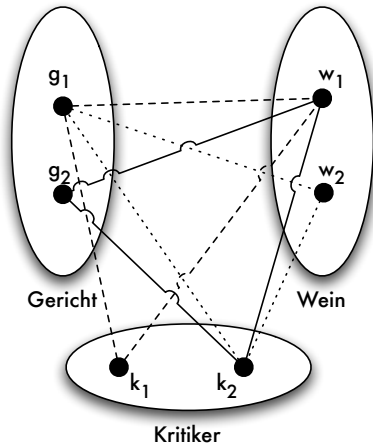
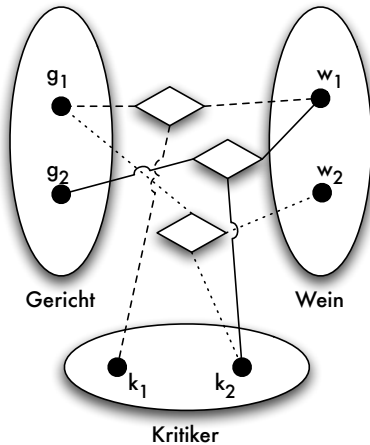
# Merkmale von Beziehungen

- **Stelligkeit** oder Grad:
  - ▶ Anzahl der beteiligten Entity-Typen
  - ▶ häufig: binär
  - ▶ Beispiel: *Lieferant **liefert** Produkt*
- **Kardinalität** oder Funktionalität:
  - ▶ Anzahl der eingehenden Instanzen eines Entity-Typs
  - ▶ Formen: 1:1, 1:n, m:n
  - ▶ stellt Integritätsbedingung dar
  - ▶ Beispiel: ***maximal 5** Produkte **pro** Bestellung*

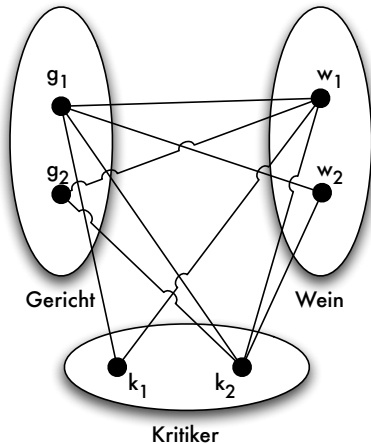
## Zwei- vs. mehrstellige Beziehungen



# Ausprägungen im Beispiel



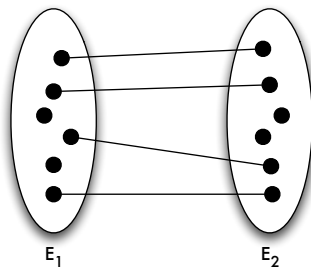
# Rekonstruktion der Ausprägungen



- $g_1 - k_1 - w_1$
- $g_1 - k_2 - w_2$
- $g_2 - k_2 - w_1$
- aber auch:  $g_1 - k_2 - w_1$

# 1:1-Beziehungen

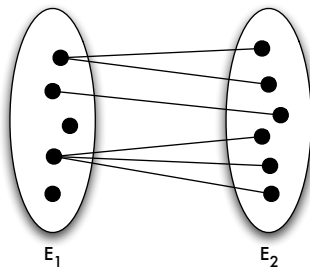
- jedem Entity  $e_1$  vom Entity-Typ  $E_1$  ist maximal ein Entity  $e_2$  aus  $E_2$  zugeordnet und umgekehrt
- Beispiele: *Prospekt **beschreibt** Produkt, Mann **ist verheiratet mit** Frau*





# 1:N-Beziehungen

- jedem Entity  $e_1$  vom Entity-Typ  $E_1$  sind beliebig viele Entities  $E_2$  zugeordnet, aber zu jedem Entity  $e_2$  gibt es maximal ein  $e_1$  aus  $E_1$
- Beispiele: *Lieferant **liefert** Produkt, Mutter **hat** Kinder*



# N:1-Beziehung

- invers zu 1:N, auch **funktionale** Beziehung
- zweistellige Beziehungen, die eine **Funktion** beschreiben:  
Jedem Entity eines Entity-Typs  $E_1$  wird maximal ein Entity eines Entity-Typs  $E_2$  zugeordnet.

$$R : E_1 \rightarrow E_2$$

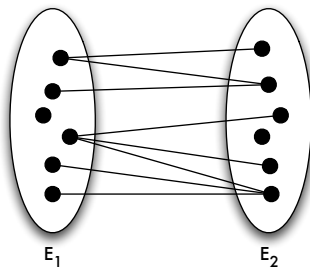


# 1:1-Beziehung

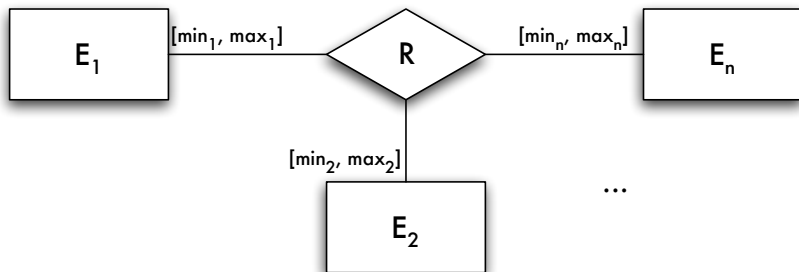


# M:N-Beziehungen

- keine Restriktionen
- Beispiel: *Bestellung* **umfasst** *Produkte*



## [min,max]-Notation



- schränkt die möglichen **Teilnahmen** von Instanzen der beteiligten Entity-Typen an der Beziehung ein, indem ein minimaler und ein maximaler Wert vorgegeben wird
- Notation für Kardinalitätsangaben an einem Beziehungstyp

$$R(E_1, \dots, E_i[\min_i, \max_i], \dots, E_n)$$

- Kardinalitätsbedingung:  $\min_i \leq |\{r \mid r \in R \wedge r.E_i = e_i\}| \leq \max_i$
- Spezielle Wertangabe für  $\max_i$  ist \*

# Kardinalitätsangaben

- $[0, *]$  legt keine Einschränkung fest (default)
- $R(E_1[0, 1], E_2)$  entspricht einer (partiellen) funktionalen Beziehung  $R : E_1 \rightarrow E_2$ , da jede Instanz aus  $E_1$  maximal einer Instanz aus  $E_2$  zugeordnet ist
- totale funktionale Beziehung wird durch  $R(E_1[1, 1], E_2)$  modelliert

# Kardinalitätsangaben: Beispiele

- partielle funktionale Beziehung

$\lceil \text{lagert\_in}(\text{Produkt}[0,1], \text{Fach}[0,3])$

„Jedes Produkt ist im Lager in einem Fach abgelegt, allerdings wird ausverkauften bzw. gegenwärtig nicht lieferbaren Produkte kein Fach zugeordnet. Pro Fach können maximal drei Produkte gelagert werden.“

- totale funktionale Beziehung

$\lceil \text{liefert}(\text{Lieferant}[0,*], \text{Produkt}[1,1])$

„Jedes Produkt wird durch genau einen Lieferant geliefert, aber ein Lieferant kann durchaus mehrere Produkte liefern.“

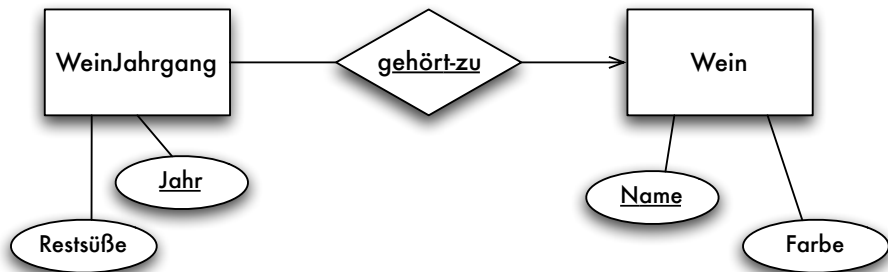
# Alternative Kardinalitätsangabe





## Abhängige Entity-Typen

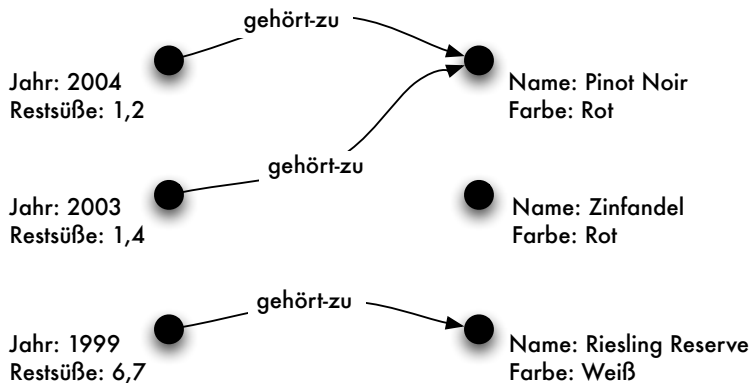
- *abhängiger Entity-Typ*: Identifikation über funktionale Beziehung



- Abhängige Entities im ER-Modell: Funktionale Beziehung als Schlüssel

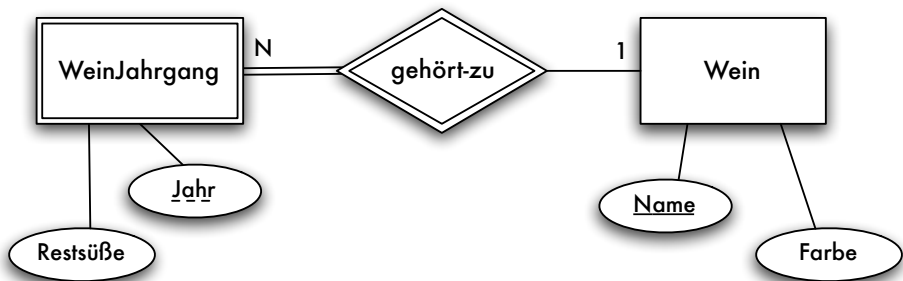
## Abhängige Entity-Typen /2

- Mögliche Ausprägung für abhängige Entities



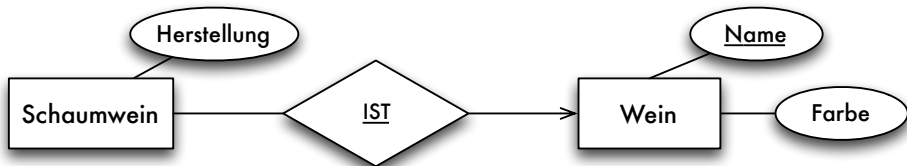
## Abhängige Entity-Typen /3

- Alternative Notation



# Die IST-Beziehung

- **Spezialisierungs-/Generalisierungsbeziehung** oder auch IST-Beziehung (engl. *is-a relationship*)
- textuelle Notation:  $E_1$  IST  $E_2$
- IST-Beziehung entspricht semantisch einer **injektiven** funktionalen Beziehung



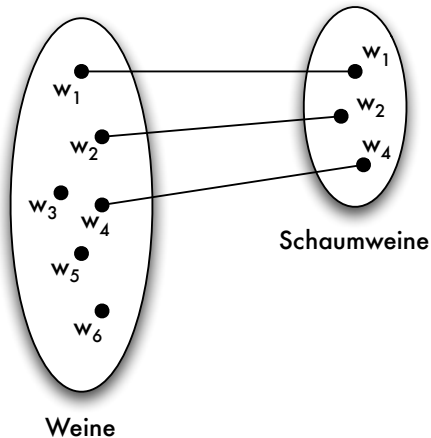
## Eigenschaften der IST-Beziehung

- Jeder Schaumwein-Instanz ist genau eine Wein-Instanz zugeordnet  
↔ Schaumwein-Instanzen werden durch die funktionale IST-Beziehung identifiziert
- Nicht jeder Wein ist zugleich ein Schaumwein
- Attribute des Entity-Typs Wein treffen auch auf Schaumweine zu: „vererbte“ Attribute

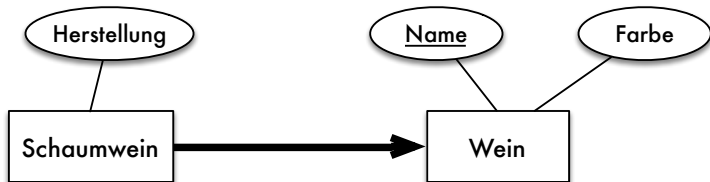
Schaumwein(Name, Farbe, Herstellung)  
*von Wein*

- nicht nur die Attributdeklarationen vererben sich, sondern auch jeweils die aktuellen Werte für eine Instanz

# Ausprägung für IST-Beziehung



## Alternative Notation für IST-Beziehung

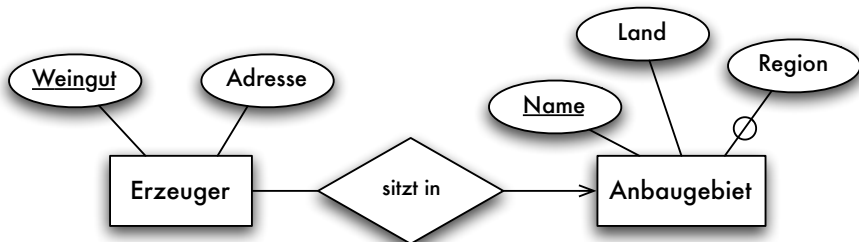


## Kardinalitätsangaben: IST

- für Beziehung  $E_1$  IST  $E_2$  gilt immer:  $\text{IST}(E_1[1, 1], E_2[0, 1])$
- Jede Instanz von  $E_1$  nimmt genau einmal an der IST-Beziehung teil, während Instanzen des Obertyps  $E_2$  nicht teilnehmen müssen
- Aspekte wie Attributvererbung werden hiervon nicht erfasst



# Optionalität von Attributen



# Konzepte im Überblick

<b>Begriff</b>	<b>Informale Bedeutung</b>
Entity	zu repräsentierende Informationseinheit
Entity-Typ	Gruppierung von Entitys mit gleichen Eigenschaften
Beziehungstyp	Gruppierung von Beziehungen zwischen Entitys
Attribut	datenwertige Eigenschaft eines Entitys oder einer Beziehung
Schlüssel	identifizierende Eigenschaft von Entitys
Kardinalitäten	Einschränkung von Beziehungstypen bezüglich der mehrfachen Teilnahme von Entitys an der Beziehung
Stelligkeit	Anzahl der an einem Beziehungstyp beteiligten Entity-Typen
funktionale Beziehung	Beziehungstyp mit Funktionseigenschaft
abhängige Entitys	Entitys, die nur abhängig von anderen Entitys existieren können
IST-Beziehung	Spezialisierung von Entity-Typen
Optionalität	Attribute oder funktionale Beziehungen als partielle Funktionen

# Zusammenfassung

- Datenbankmodell, Datenbankschema, Datenbank(instanz)
- Entity-Relationship-Modell
- Weitere Konzepte im ER-Modell
- *Basis: Kapitel 3 von [SSH10]*

# Kontrollfragen

- Was definiert ein Datenbankmodell? Was unterscheidet Modell und Schema?
- Welche Konzepte definiert das ER-Modell?
- Durch welche Eigenschaften sind Beziehungstypen charakterisiert?
- Was unterscheidet abhängige Entity-Typen von normalen Entity-Typen?

