

# Datenbanken 1

## Das Relationale Modell

Nikolaus Augsten

nikolaus.augsten@sbg.ac.at  
FB Computerwissenschaften  
Universität Salzburg



<http://dbresearch.uni-salzburg.at>

Sommersemester 2020

Version 24. März 2020

- 1 Das Relationale Modell
  - Schema, Relation, und Datenbank
  - Integritätsbedingungen
- 2 Abbildung ER-Schema auf Relationales Modell

# Literatur und Quellen

Lektüre zum Thema “Relationales Modell”:

- Kapitel 3 (3.1-3.3) aus Kemper und Eickler: Datenbanksysteme: Eine Einführung. 9. Auflage, Oldenbourg Verlag, 2013.

## Literaturquellen

- Elmasri and Navathe: Fundamentals of Database Systems. Fourth Edition, Pearson Addison Wesley, 2004.
- Silberschatz, Korth, and Sudarashan: Database System Concepts, McGraw Hill, 2006.

Danksagung Die Vorlage zu diesen Folien wurde entwickelt von:

- Michael Böhlen, Universität Zürich, Schweiz
- Johann Gamper, Freie Universität Bozen, Italien

# Inhalt

- 1 Das Relationale Modell
  - Schema, Relation, und Datenbank
  - Integritätsbedingungen
- 2 Abbildung ER-Schema auf Relationales Modell

# Das Relationale Modell/1

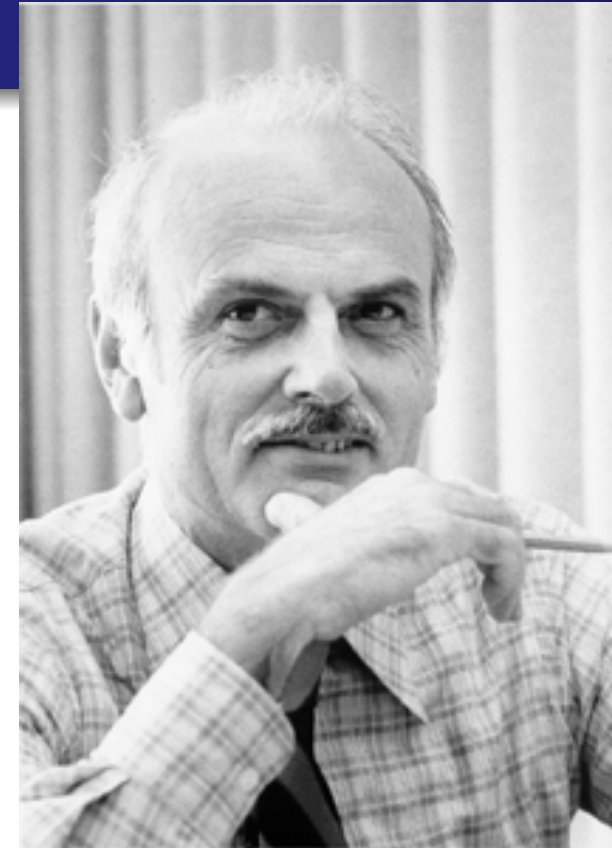
- **Relationale Modell:**
  - logisches Datenmodell
  - basiert auf Relationen
- **Relation:** mathematisches Konzept, das auf Mengen basiert.
- Die **Stärke** des relationalen Modells ist die **formale Grundlage** durch Relationen (und Mengen).
- In der **Praxis** wird der **SQL Standard** verwendet. Der SQL Standard unterscheidet sich vom formalen Modell in einigen Punkten (wir gehen später auf diese Unterschiede ein).

# Das Relationale Modell/2

- Das relationale Modell wurde von [E. Codd von IBM Research](#) in folgendem Artikel eingeführt:  
*A Relational Model for Large Shared Data Banks, Communications of the ACM, June 1970*
- Dieser Artikel hat das Feld der Datenbanksysteme revolutioniert.
- Codd erhielt hierfür den [ACM Turing Award](#).

# Das Relationale Modell/3

- Edgar Codd, a mathematician and IBM Fellow, is best known for creating the relational model for representing data that led to today's 12 billion database industry.
- Codd's basic idea was that **relationships** between data items **should be based on the item's values**, and not on separately specified linking or nesting.
- The idea of relying only on value-based relationships was quite a radical concept at that time, and many people were skeptical. They didn't believe that **machine-made relational queries** would be able to perform as well as **hand-tuned programs** written by expert human navigators.



[http://www-03.ibm.com/ibm/history/exhibits/builders/builders\\_codd.html](http://www-03.ibm.com/ibm/history/exhibits/builders/builders_codd.html)

# Schema

- $sch(R) = [A_1, A_2, \dots, A_n]$  ist das **Schema** der Relation.
- Eckige Klammern [...] werden für eine Liste von Werten verwendet; eine Liste ist geordnet.
- $R$  ist der **Name** der Relation.
- $A_1, A_2, \dots, A_n$  sind die **Attribute**.
- **Kurzschreibweise**: Für die Definition einer Relation  $R$  mit Schema  $sch(R) = [A_1, A_2, \dots, A_n]$  schreiben wir kurz:

$$R[A_1, A_2, \dots, A_n]$$

- **Beispiel**: Für die Relation  $Kunden[KundenName, KundenStrasse, KundenOrt]$  gilt  $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$



# Die Relation Kunden

- Schema:  $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$

Kunden

<b>KundenName</b>	<b>KundenStrasse</b>	<b>KundenOrt</b>
Meier	Zeltweg	Brugg
Steger	Ringstr	Aarau
Marti	Seeweg	Brugg
Kurz	Marktplatz	Luzern
Egger	Weststr	Brugg
Staub	Bahnhofstr	Brugg
Gamper	Bahnhofstr	Chur
Ludwig	Baugasse	Brugg
Wolf	Bahnhofstr	Brugg
Koster	Magnolienweg	Brugg
Kunz	Fliedergasse	Brugg
Pauli	Murtenstr	Biel

# Domäne

- Eine **Domäne** ist eine Menge von atomaren Werten.
  - Beispiel: Alter einer Person ist eine positive Ganzzahl.
- Zu jeder Domäne gehört ein **Datentyp** (oder Format):
  - Telefonnummer hat Format: Odd ddd dd dd, wobei d eine Ziffer ist.
  - Für ein Datum existieren verschiedene Formate: z.B. yyyy-mm-dd oder dd.mm.yyyy
- Der **reservierte Wert null** gehört zu jeder Domäne:
  - wird für fehlende Werte verwendet
  - Nullwerte machen die Definition von Operationen komplexer

# Attribute

- **Attributwert:** Attribut nimmt für jedes Tupel einen Wert an
  - mögliche Werte durch Domäne bestimmt
  - $dom(A)$  ist die Domäne von Attribut  $A$
- **Atomar:** Attributwerte müssen atomar sein
  - also “einfach” im Sinne des ER-Modells
  - zusammengesetzte oder mehrwertige Attribute sind nicht erlaubt
  - “Pink Floyd” ist atomar, “Pink Floyd – Wish you were here” ist nicht atomar
- **Attributname:** spezifizieren Rolle der entsprechenden Domäne in Relation:
  - Name ist eindeutig innerhalb einer Relation
  - wird verwendet, um die Werte dieses Attributs zu interpretieren
- **Beispiel:** Die Domäne *Datum* wird für die Attribute *Rechnungsdatum* und *Zahlungstermin* mit unterschiedlichen Bedeutungen verwendet.
  - $dom(Rechnungsdatum) = Datum$
  - $dom(Zahlungstermin) = Datum$

# Tupel

- Ein **Tupel** ist eine geordnete Menge (d.h. eine Liste) von Werten
- Eckige Klammern [...] werden verwendet um Tupel darzustellen
- Jeder Wert eines Tupels muss aus der entsprechenden Domäne stammen
- **Beispiel:** Tupel der Relation *Kunden*
  - Schema:  $sch(Kunden) = [KundenName, KundenStrasse, KundenOrt]$
  - Tupel:  $[Meier, Zeltweg, Brugg]$

# Instanz (Ausprägung)

- Der Name einer Relation  $R$  wird auch als Bezeichner für die Instanz einer Relation verwendet
- Instanz einer Relation  $R$  mit Schema  $sch(R) = [A_1, A_2, \dots, A_n]$  ist eine Untermenge des Kreuzprodukts der Domänen der Attribute:

$$R \subseteq D_1 \times D_2 \times \dots \times D_n$$

$R$  ist also eine Menge von Tupeln  $[v_1, v_2, \dots, v_n]$ , sodass jedes  $v_i \in D_i$

- Geschweifte Klammern  $\{\dots\}$  werden für Mengen verwendet
- Beispiel:

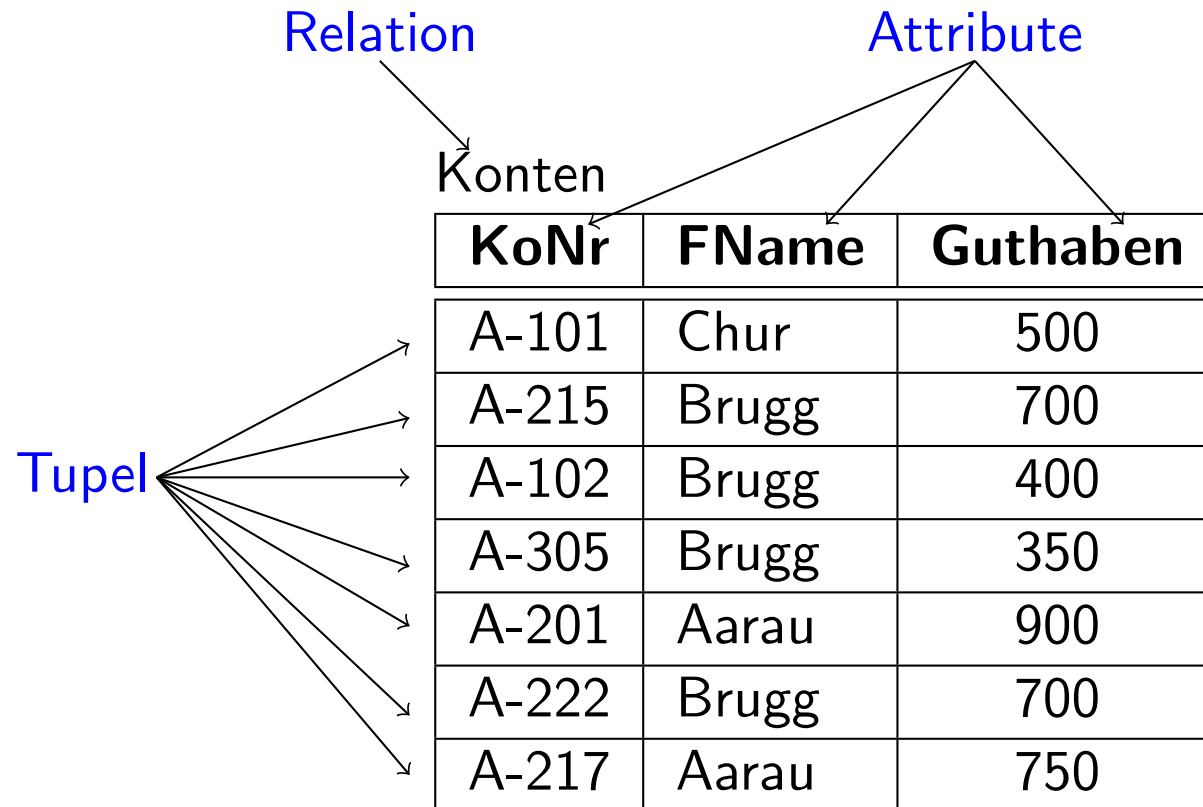
$D_1 = dom(KuName) = \{Ludwig, Koster, Marti, Wolf, \dots\}$

$D_2 = dom(KuStrasse) = \{Bahnhofstr, Baugasse, Seeweg, \dots\}$

$D_3 = dom(KuOrt) = \{Brugg, Luzern, Chur, \dots\}$

$R = \{ [Ludwig, Bahnhofstr, Brugg], [Koster, Baugasse, Brugg],$   
 $[Marti, Seeweg, Brugg], [Wolf, Weststr, Brugg] \}$   
 $\subseteq D_1 \times D_2 \times D_3$

# Beispiel einer Relation



# Eigenschaften von Relationen

- Relationen sind ungeordnet, d.h., die Ordnung der Tupel ist nicht relevant.
- Die Attribute eines Schemas  $sch(R) = [A_1, \dots, A_n]$  und die Werte in einem Tuple  $t = [v_1, \dots, v_n]$  sind geordnet.

Konten

KoNr	FName	Guthaben
A-101	Chur	500
A-215	Brugg	700
A-102	Brugg	400
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-217	Aarau	750

=

Konten

KoNr	FName	Guthaben
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-102	Brugg	400
A-217	Aarau	750
A-101	Chur	500
A-215	Brugg	700

# Integrierte Übung 3.1

1. Ist  $R = \{[Tom, 27, ZH], [Bob, 33, Rome, IT]\}$  eine Relation?
2. Was ist der Unterschied zwischen einer Menge und einer Relation?  
Geben Sie ein Beispiel, das den Unterschied illustriert.



# Integrierte Übung 3.2

1. Illustrieren Sie die folgenden Relationen graphisch:

$$\text{sch}(R) = [X, Y]; R = \{[1, a], [2, b], [3, c]\}$$

$$\text{sch}(S) = [A, B, C]; S = \{[1, 2, 3]\}$$

2. Bestimmen Sie die folgenden Objekte:

- Das 2. Attribut der Relation  $R$ ?
- Das 3. Tupel der Relation  $R$ ?
- Das Tuple in der Relation  $R$  mit dem kleinsten Wert von Attribut  $X$ ?

# Datenbank

- Eine **Datenbank** ist eine **Menge von Relationen**.
- **Beispiel:** Die Informationen eines Unternehmens werden in mehrere Teile aufgespaltet:
  - *Konten*: speichert Informationen über Konten
  - *Kunde*: speichert Informationen über Kunden
  - *Kontoinhaber*: speichert welche Kunden welche Konten besitzen
- Warum nicht alle Informationen in eine Relation speichern?
  - Beispiel:  $sch(Bank) = [KoNr, Guthaben, KuName, \dots]$
  - **Redundanz:** Wiederholung von Informationen, z.B. zwei Kunden mit demselben Konto
  - **Nullwerte:** z.B. für einen Kunden ohne Konto

## Datenbank mit Relationen Konten und Kontoinhaber

Konten

<b>KoNr</b>	<b>FName</b>	<b>Guthaben</b>
A-101	Chur	500
A-215	Brugg	700
A-102	Brugg	400
A-305	Brugg	350
A-201	Aarau	900
A-222	Brugg	700
A-217	Aarau	750

Kontoinhaber

<b>KName</b>	<b>KoNr</b>
Staub	A-102
Gamper	A-101
Gamper	A-201
Ludwig	A-217
Wolf	A-222
Koster	A-215
Kunz	A-305
Mair	A-101

# Integrierte Übung 3.3

1. Welche Art von Objekt ist  $X = \{\{[3]\}\}$  im relationalen Modell?

2. Sind DB1 und DB2 identische Datenbanken?

$$DB1 = \{\{[1, 5], [2, 3]\}, \{[4, 4]\}\}$$

$$DB2 = \{\{[4, 4]\}, \{[2, 3], [1, 5]\}\}$$

# Zusammenfassung des relationalen Modells

- Eine **Domäne**  $D$  ist eine Menge von atomaren Werten.
  - Telefonnummern, Namen, Noten, Geburtstage, Institute
  - jede Domäne beinhaltet den reservierten Wert `null`
- Zu jeder Domäne wird ein **Datentyp** oder Format spezifiziert.
  - 5-stellige Zahlen, yyyy-mm-dd, Zeichenketten
- Ein **Attribut**  $A_i$  beschreibt die Rolle einer Domäne innerhalb eines Schemas.
  - TelephonNr, Alter, Institutsname
- Ein **Schema**  $sch(R) = [A_1, \dots, A_n]$  besteht aus einer Liste von Attributen.
  - $sch(\text{Angestellte}) = [\text{Name}, \text{Institut}, \text{Lohn}]$ ,
  - $sch(\text{Institute}) = [\text{InstName}, \text{Leiter}, \text{Adresse}]$
- Ein **Tupel**  $t$  ist eine Liste von Werten  $t = [v_1, \dots, v_n]$  mit  $v_i \in dom(A_i)$ .
  - $t = [\text{Tom}, \text{SE}, 23K]$
- Eine **Relation**  $R \subseteq D_1 \times \dots \times D_n$  mit dem Schema  $sch(R) = [A_1, \dots, A_n]$  ist eine Menge von n-stelligen Tupeln.
  - $R = \{[\text{Tom}, \text{SE}, 23K], [\text{Lene}, \text{DB}, 33K]\} \subseteq \text{NAMEN} \times \text{INSTITUTE} \times \text{INTEGER}$
- Eine **Datenbank**  $DB$  ist eine Menge von Relationen.
  - $DB = \{R, S\}$
  - $R = \{[\text{Tom}, \text{SE}, 23K], [\text{Lene}, \text{DB}, 33K]\}$
  - $S = \{[\text{SE}, \text{Tom}, \text{Boston}], [\text{DB}, \text{Lena}, \text{Tucson}]\}$

# Integritätsbedingungen

- **Integritätsbedingungen** (constraints) sind Einschränkungen auf den Daten, die alle Instanzen der Datenbank erfüllen müssen.
- **Klassen von Integritätsbedingungen** im relationalen Modell :
  - Schlüssel
  - Domänenintegrität
  - Referentielle Integrität
- Integritätsbedingungen garantieren eine gute **Datenqualität**.

## Schlüssel/1

- $K \subseteq R$  ist eine Teilmenge der Attribute von  $R$
- $K$  ist ein **Superschlüssel** von  $R$  falls die Werte von  $K$  ausreichen um ein Tupel jeder möglichen Relation  $R$  eindeutig zu identifizieren.
  - Mit “jeder möglichen” meinen wir eine Relation, die in der Miniwelt, die wir modellieren, existieren könnte.
  - Beispiel:  $\{KuName, KuStrasse\}$  und  $\{KuName\}$  sind Superschlüssel von *Kunde*, falls keine zwei Kunden den gleichen Namen haben können.

KuName	KuStrasse
N. Jeff	Binzmühlestr
N. Jeff	Hochstr

KuName ist *kein* Schlüssel

KuName	KuStrasse
N. Jeff	Binzmühlestr
T. Hurd	Hochstr

KuName ist ein Schlüssel

# Schlüssel/2

- $K$  ist ein **Kandidatschlüssel** falls  $K$  minimal ist

Beispiel:

- $\{KuName\}$  ist ein Kandidatschlüssel für *Kunde* weil diese Menge ein Superschlüssel ist und keine Untermenge ein Superschlüssel ist.
  - $\{KuName, KuStrasse\}$  ist kein Kandidatschlüssel weil eine Untermenge, nämlich  $\{KuName\}$ , ein Superschlüssel ist.
- 
- **Primärschlüssel:** ein Kandidatschlüssel der verwendet wird um Tupel in einer Relation zu identifizieren.
    - Als Primärschlüssel sollte ein Attribut ausgewählt werden, dessen Wert sich nie ändert (oder zumindest sehr selten).
    - Beispiel: *email* ist eindeutig und ändert sich selten



# Domänenintegrität

- Die **Domänenintegrität** garantiert, dass alle Attributwerte aus der entsprechenden Domäne stammen.
- **Nullwerte:** sind standardmäßig erlaubt da Teil der Domäne
- **Primärschlüssel** dürfen **nicht null** sein
  - falls der Primärschlüssel aus mehreren Attributen besteht darf keines dieser Attribute null sein
  - andere Attribute der Relation, selbst wenn sie nicht zum Primärschlüssel gehören, können ebenfalls Nullwerte verbieten

ID	Name	KuStrasse
1	N. Jeff	Binzmühlestr
null	T. Hurd	Hochstr

ID kann nicht Primärschlüssel sein

ID	Name	KuStrasse
1	N. Jeff	Binzmühlestr
2	T. Hurd	Hochstr

ID kann Primärschlüssel sein

# Referentielle Integrität

- **Fremdschlüssel:** Attribute im Schema einer Relation, die Primärschlüssel einer anderen Relation sind.
  - Beispiel: *KuName* und *KoNr* der Relation *Kontoinhaber* sind Fremdschlüssel von *Kunde* bzw. *Konten*.
- **Rekursion:** Nicht-Primärschlüssel Attribute können auch Fremdschlüssel zum Primärschlüssel in derselben Relation sein.
- **Erlaubte Werte** für Fremdschlüssel:
  - Werte, die als Primärschlüssel in der referenzierten Relation vorkommen
  - null Werte (alle oder kein Attribut des Fremdschlüssels)
- **Graphischen Darstellung** eines Schemas: gerichteter Pfeil vom Fremdschlüsselattribut zum Primärschlüsselattribut.

ID	KuName	KuStrasse
1	N. Jeff	2
2	T. Hurd	4

StrassenNr	Strasse
2	Binzmühlestr
3	Hochstr

KuStrasse kann kein Fremdschlüssel sein weil StrassenNr 4 nicht existiert.

# Integrierte Übung 3.4

- Bestimmen Sie die Schlüssel der Relation  $R$ :

$R$

X	Y	Z
1	2	3
1	4	5
2	2	2

# Integrierte Übung 3.5

- Bestimmen Sie mögliche Superschlüssel, Kandidatschlüssel, Primärschlüssel und Fremdschlüssel für die Relationen  $R$  und  $S$ :

R			S	
A	B	C	D	E
a	d	e	d	a
b	d	c	e	a
c	e	e	a	a

mögliche Superschlüssel:

mögliche Kandidatschlüssel:

mögliche Primärschlüssel:

mögliche Fremdschlüssel:

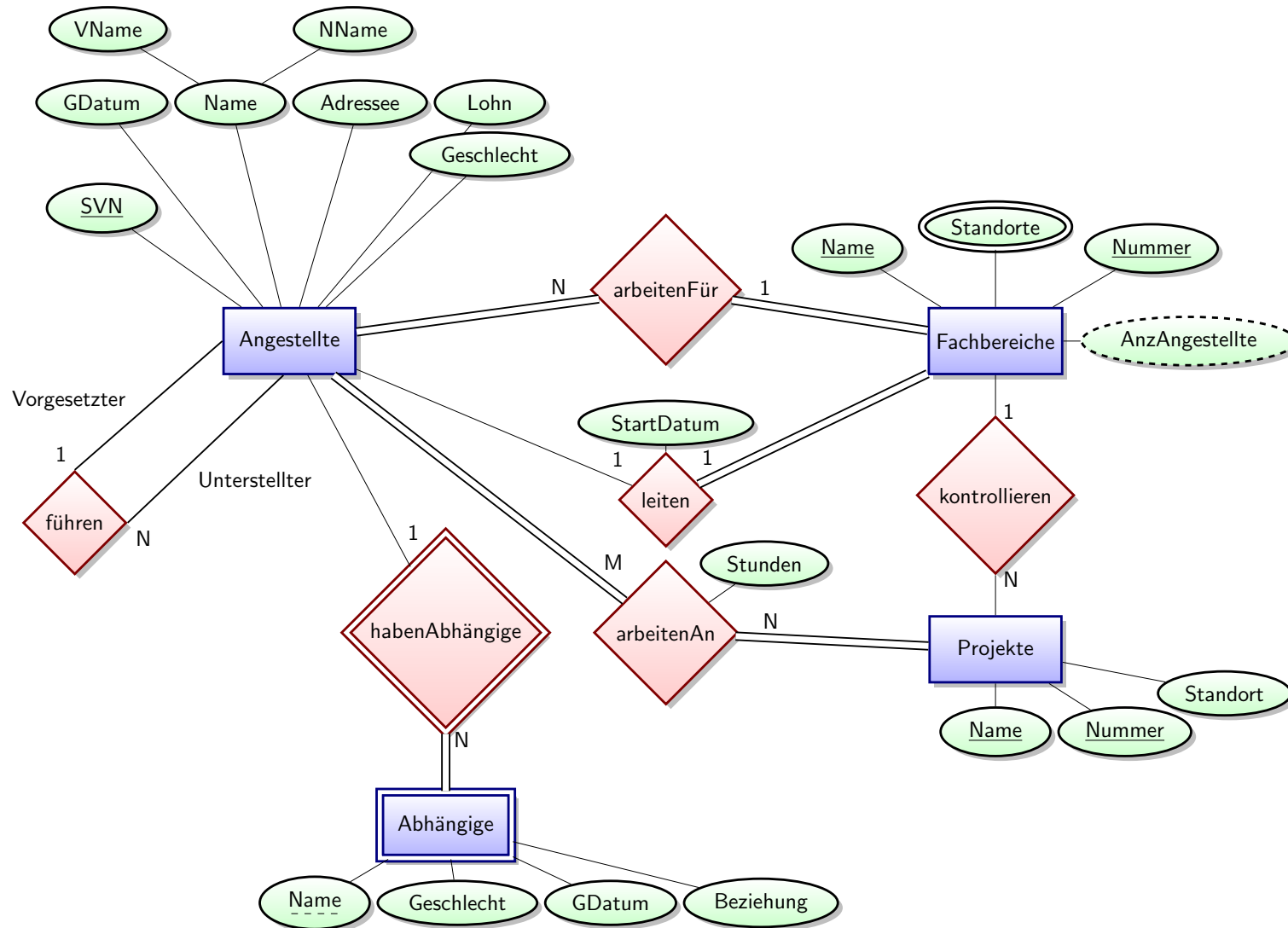
# Inhalt

- 1 Das Relationale Modell
  - Schema, Relation, und Datenbank
  - Integritätsbedingungen
  
- 2 Abbildung ER-Schema auf Relationales Modell

# Algorithmus ER-Schema → Relationales Modell

- Algorithmus um ein konzeptionelles ER-Schema (fast) automatisch in ein relationales Schema abzubilden.
  - Schritt 1: unabhängige Entitätstypen
  - Schritt 2: existenzabhängige Entitätstypen
  - Schritt 3: Beziehungstypen
  - Schritt 4: mehrwertige Attribute
  - Schritt 5: n-wertigen Beziehungstypen
  - Schritt 6: Spezialisierung/Generalisierung

# Beispiel: ER Schema der NAWI Datenbank



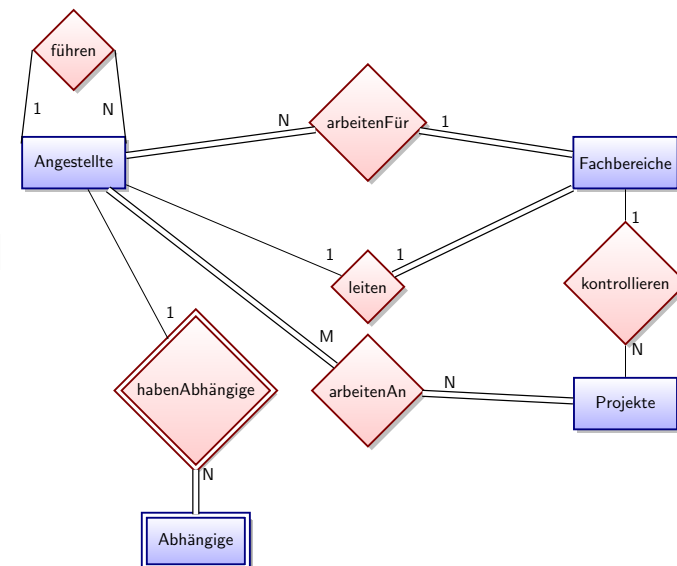
# Schritt 1: Abbildung unabhängiger Entitätstypen

- (a) **Entitätstyp**: Für jeden unabhängigen Entitätstypen  $E$  erstellen wir eine Relation  $R$ .
- (b) **Attribute**: Die Attribute von  $R$  sind
  - alle einfachen Attributen von  $E$
  - alle einfachen Komponenten von zusammengesetzten Attributen
- (c) **Primärschlüssel**: Ein Schlüsselattribut von  $E$  wird als Primärschlüssel für  $R$  ausgewählt.
  - ➔ Falls der ausgewählte Schlüssel von  $E$  zusammengesetzt ist, besteht der Primärschlüssel aus allen einfachen Komponenten.



# Beispiel: Abbildung unabhängiger Entitätstypen

- **Beispiel:** Wir erstellen Relationen Angestellte, Fachbereiche, Projekte.
  - SVN, FNummer, und PNummer sind die Primärschlüssel



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn]

Fachbereiche[FName, FNummer]

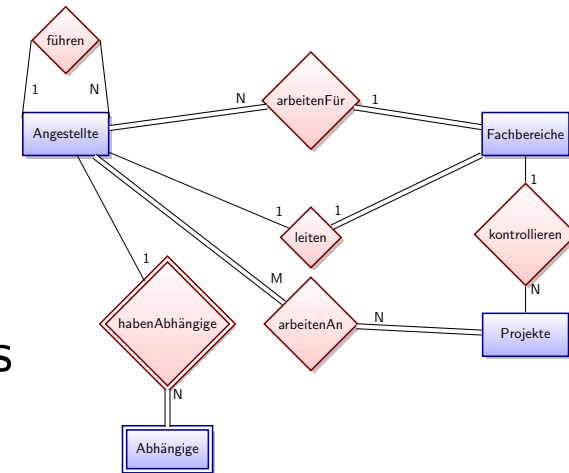
Projekte[PName, PNummer, PStandort]

## Schritt 2: Abbildung existenzabhängiger Entitätstypen

- (a) **Existenzabhängiger Entitätstyp:** Für jeden existenzabhängigen Entitätstypen  $W$  mit übergeordnetem Entitätstypen  $E$  erstellen wir eine Relation  $R$ .
- (b) **Attribute** von  $R$  sind alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute von  $W$ .
- (c) **Fremdschlüssel:** Der Primärschlüssel der Relation des übergeordneten Entitätstypen  $E$  wird als Fremdschlüssel zu  $R$  hinzugefügt.
- (d) **Primärschlüssel** von  $R$  besteht aus der *Kombination* der
  - Primärschlüssel der übergeordneten Entitätstypen
  - des partiellen Schlüssels des existenzabhängigen Entitätstypen

# Beispiel: Abbildung existenzabhängiger Entitätstypen

- **Beispiel:** Der existenzabhängigen Entitätstypen **Abhängige** wird auf Relation **Abhängige** abgebildet.
- Primärschlüssel SVN von Angestellte wird als **Fremdschlüssel** zu Relation **Abhängige** hinzugefügt (umbenannt auf AngSVN).
- Der **Primärschlüssel** von Abhängige ist die Kombination { **AngSVN**, **AbhName** }, weil **AbhName** ein partieller Schlüssel von **Abhängige** ist.



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn]

Fachbereiche[FName, FNummer]

Projekte[PName, PNummer, PStandort]

Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]

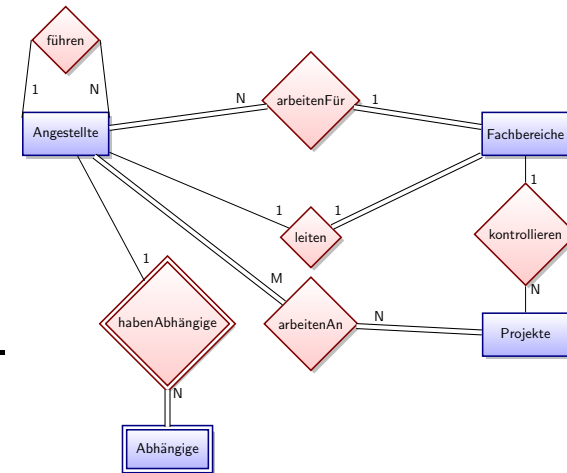
## Schritt 3: Abbildung von Beziehungen

**Drei mögliche Ansätze** für Beziehung zwischen Entitätstypen  $S$  und  $T$ :

1. **Zusammengefasste Relationen:** nur für **1:1**
  - beteiligte Entitätstypen werden in einzige Relation zusammengelegt
  - keine Nullwerte falls  $S$  und  $T$  totale Beziehung eingehen
2. **Fremdschlüssel:** 1:1, **1:N**
  - eine der beteiligten Entitäten wird ausgewählt, z.B.  $S$  ( $N$ -Seite im Falle von 1:N)
  - Primärschlüssel von  $T$  wird als Fremdschlüssel zu  $S$  hinzugefügt
  - keine Nullwerte falls  $S$  totale Beziehung eingeht
3. **Neue Beziehungsrelation:** 1:1, 1:N, **M:N**
  - neue Relation  $R$  mit den Primärschlüsseln von  $S$  und  $T$  als Fremdschlüssel
  - Primärschlüssel:
    - 1 : 1-Beziehung: einer der beiden Fremdschlüssel
    - 1 :  $N$ -Beziehung: Fremdschlüssel der  $N$ -Seite
    - $M$  :  $N$ -Beziehung: beide Fremdschlüssel
  - keine Nullwerte

# Beispiel: Abbildung von 1:1 Beziehungstyp

- **Beispiel:** Der 1:1 Beziehungstyp **leiten** wird mithilfe eines Fremdschlüssels abgebildet. Fachbereiche übernimmt die Rolle von S, weil die Teilnahme in der Beziehung total ist.



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn]

Fachbereiche[FName, FNummer, **LeiterSVN**, **StartDatum**]

Projekte[PName, PNummer, PStandort]

Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]

## Integrierte Übung 3.6

Illustrieren Sie die Probleme die auftreten, wenn der Beziehungstyp **leiten**

- a) durch einen Fremdschlüssel in der Relation Angestellte abgebildet wird
- b) durch Zusammenfassen der Relationen Angestellte und Fachbereiche abgebildet wird

# Beispiel: Abbildung von 1:N Beziehungstyp

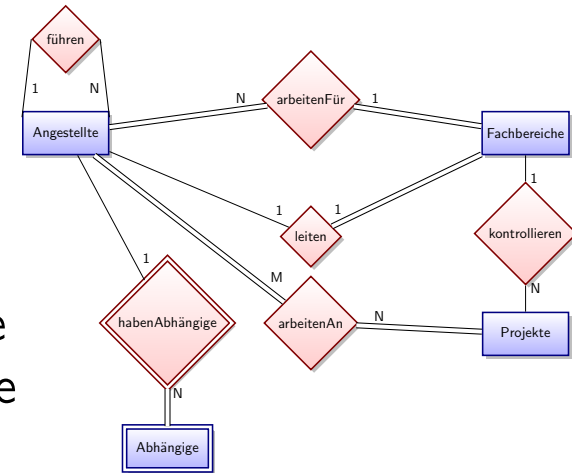
- **Beispiel:** Abbildung des N:1 Beziehungstyps

Angestellte **arbeitenFür** Fachbereiche:

- Angestellte entspricht der Relation  $S$ .
- Primärschlüssel FNummer von Fachbereiche wird Fremdschlüssel der Relation Angestellte

- Weitere 1:N Beziehungstypen:

- **Angestellte/Vorgesetzte führen Angestellte/Unterstellte:** Primärschlüssel von Angestellte als Fremdschlüssel VorgSVN zu Angestellte hinzufügen.
- **Fachbereiche kontrollieren Projekte:** Primärschlüssel von Fachbereiche als Fremdschlüssel zu Projekte hinzufügen.



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn, **VorgSVN**, **FNummer**]

Fachbereiche[FName, FNummer, LeiterSVN, StartDatum]

Projekte[PName, PNummer, PStandort, **FNummer**]

Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]

# Wie werden Attribute von Beziehungstypen abgebildet?

- Beziehung zwischen Relationen  $S$  und  $T$  soll abgebildet werden.
- **Zusammengefasste Relationen:** nur 1:1
  - $S$  und  $T$  verschmelzen zu  $R$
  - Attribute<sup>1</sup> des Beziehungstypen werden als Attribute zu  $R$  hinzugefügt
- **Fremdschlüssel:** 1:1, 1:N
  - $S$  erhält Fremdschlüssel ( $N$ -Seite im Falle von 1:N)
  - Attribute<sup>1</sup> des Beziehungstypen werden als Attribute zu  $S$  hinzugefügt
- **Neue Beziehungsrelation:** 1:1, 1:N, M:N
  - neue Relation  $R$  wird erstellt
  - Attribute<sup>1</sup> des Beziehungstypen werden als Attribute zu  $R$  hinzugefügt

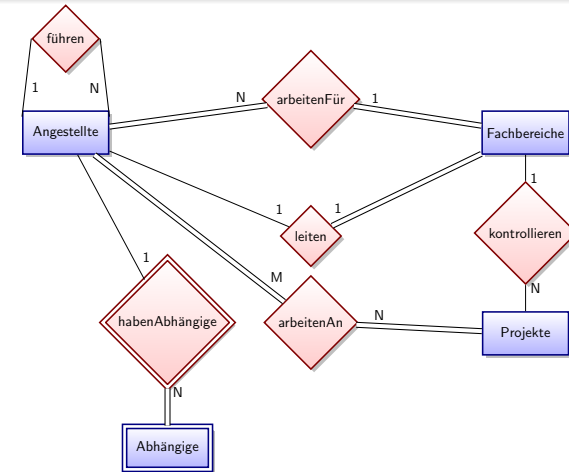
---

<sup>1</sup>alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute



# Beispiel: Abbildung von M:N Beziehungstyp mit Attributen

- **Beispiel:** Für den M:N Beziehungstyp **arbeitenAn** erstellen wir eine Relation **ArbeitenAn**.
- Die Primärschlüssel der Relationen **Projekte** und **Angestellte** werden als **Fremdschlüssel** zur Relation **ArbeitenAn** hinzugefügt.
- **Attribut *Stunden*** der Relation **ArbeitenAn** bildet das gleichnamige Attribut des Beziehungstypen ab.
- Der **Primärschlüssel** von **ArbeitenAn** ist die Kombination der Fremdschlüssel: { **AngSVN**, **PNummer** }.



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn, VorgSVN, FNummer]

Fachbereiche[FName, FNummer, LeiterSVN, StartDatum]

Projekte[PName, PNummer, PStandort, FNummer]

Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]

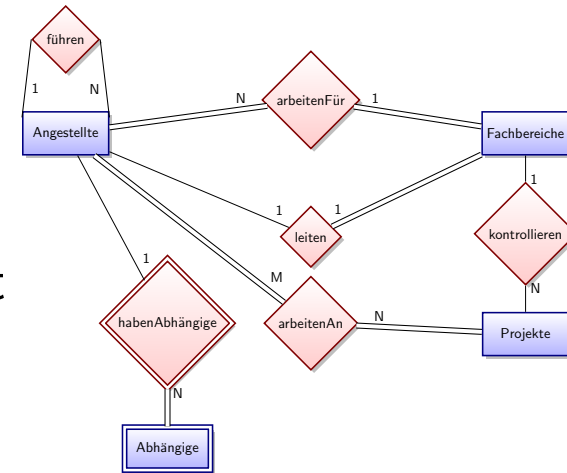
**ArbeitenAn**[AngSVN, PNummer, **Stunden**]

## Schritt 4: Abbildung mehrwertiger Attribute

- **Neue Relation:** Für jedes mehrwertige Attribut  $A$  erstellen wir eine neue Relation  $R$ .
- **Attribute:** Das mehrwertige Attribut  $A$  wird zur Relation  $R$  als (einfaches) Attribut hinzugefügt. Falls das mehrwertige Attribut  $A$  zusammengesetzt ist, werden alle einfachen Komponenten von  $A$  als (einfache) Attribute hinzugefügt.
- **Fremdschlüssel:** Primärschlüssel  $K$  der Relation, die den Entitäts- oder Beziehungstyp von  $A$  abbildet.
- **Primärschlüssel:** Kombination von  $A$  und  $K$ . Falls das mehrwertige Attribut zusammengesetzt ist, sind alle einfachen Komponenten Teil des Primärschlüssels.

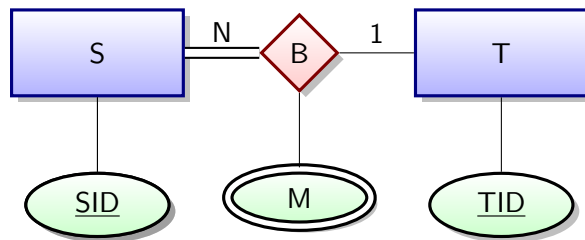
# Beispiel 1: Abbildung mehrwertiger Attribute

- **Beispiel:** das mehrwertige Attribut **Standorte** des Entitätstyps **Fachbereiche**.
- Eine **neue Relation** FBStandorte mit Attribut Standort wird erstellt.
- FNummer der Relation Fachbereiche ist **Fremdschlüssel** in FBStandorte.
- Der **Primärschlüssel** von FBStandorte sind die Attribute { FNummer, Standort }.



Angestellte[VName, NName, SVN, GDatum, Adresse, Geschlecht, Lohn, VorgSVN, FNummer]  
 Fachbereiche[FName, FNummer, LeiterSVN, StartDatum]  
 Projekte[PName, PNummer, PStandort, FNummer]  
 Abhängige[AngSVN, AbhName, Geschlecht, GDatum, Beziehung]  
 ArbeitenAn[AngSVN, PNummer, Stunden]  
 FBStandorte[FNummer, Standort]

## Beispiel 2: Abbildung mehrwertiger Attribute



- **Beispiel:** mehrwertiges Attribut **M** der 1:N Beziehung **B**
- 1:N Beziehung wird als **Fremdschlüssel** in **S** modelliert
- mehrwertiges Attribut wird durch **neue Relation MB** modelliert

S[SID, TID]

T[TID]

**MB[SID, M]**

# Beispiel: Vollständige NAWI Datenbank

Angestellte

VName	NName	<u>SVN</u>	GDatum	Adresse	Geschlecht	Lohn	VorgSVN	FNummer
-------	-------	------------	--------	---------	------------	------	---------	---------

Fachbereiche

FName	<u>FNummer</u>	LeiterSVN	StartDatum
-------	----------------	-----------	------------

FBStandorte

<u>FNummer</u>	<u>Standort</u>
----------------	-----------------

Projekte

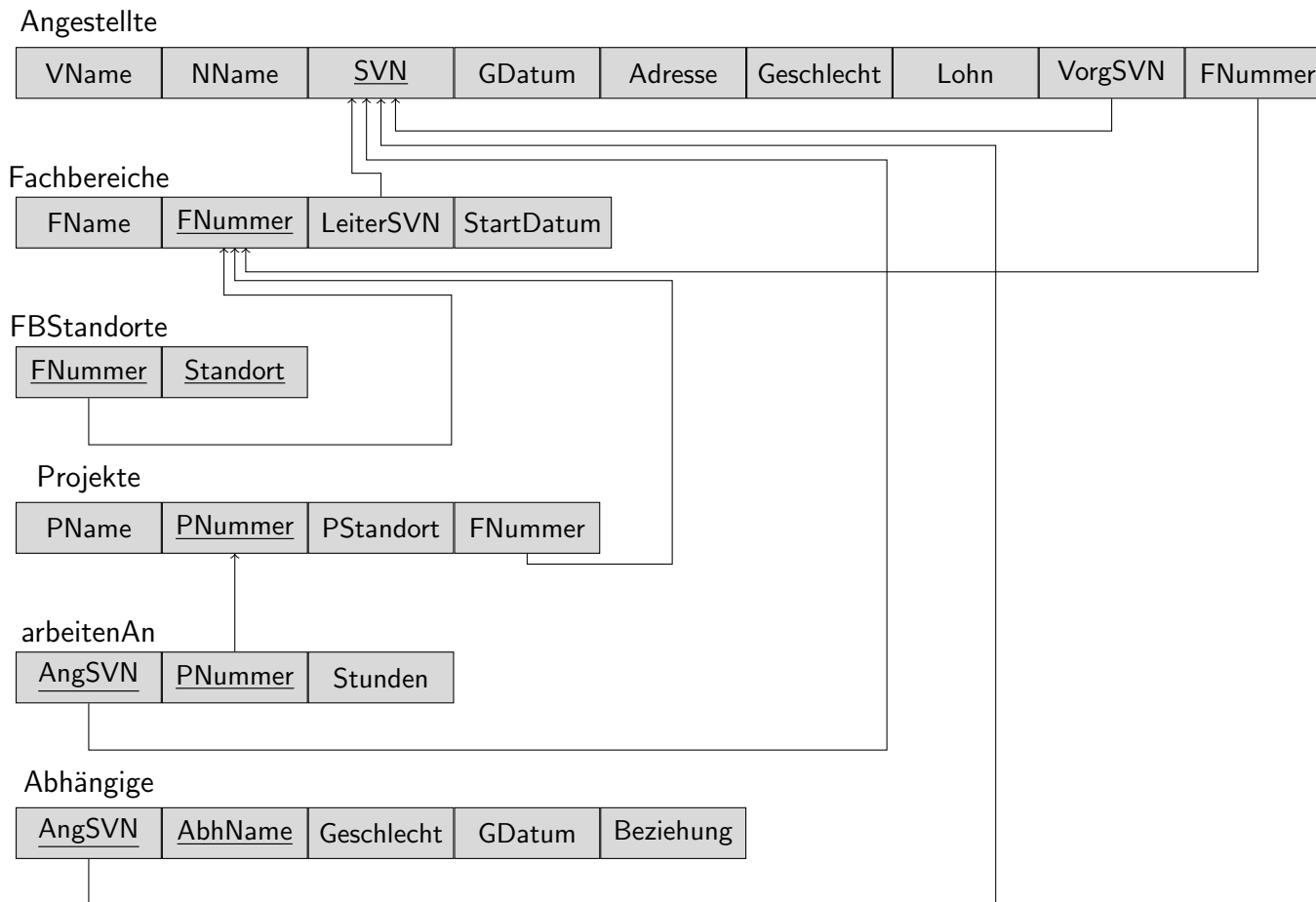
PName	<u>PNummer</u>	PStandort	FNummer
-------	----------------	-----------	---------

arbeitenAn

<u>AngSVN</u>	<u>PNummer</u>	Stunden
---------------	----------------	---------

Abhängige

<u>AngSVN</u>	<u>AbhName</u>	Geschlecht	GDatum	Beziehung
---------------	----------------	------------	--------	-----------



## Schritt 5: Abbildung von $n$ -wertigen Beziehungstypen.

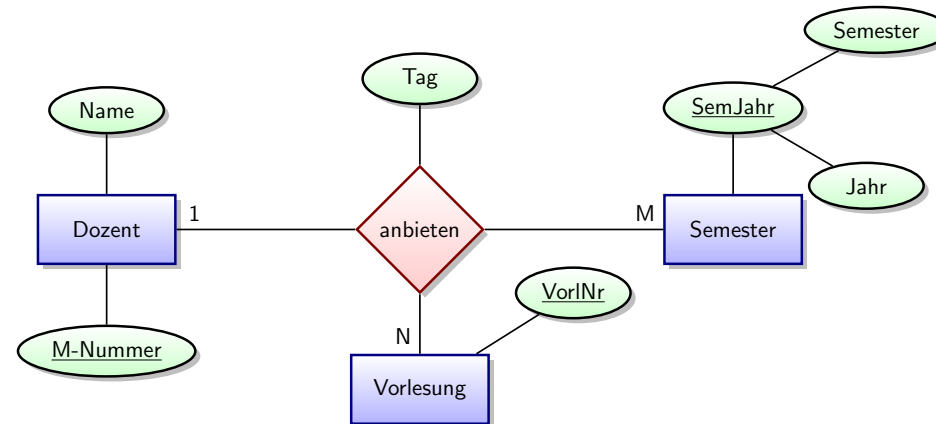
- **Neue Relation:** Für jeden  $n$ -wertigen Beziehungstypen ( $n > 2$ ) erstellen wir eine neue Relation  $R$ .
- **Fremdschlüssel:** Die Primärschlüssel der Relationen der involvierten Entitätstypen sind Fremdschlüssel in  $R$ .
- **Primärschlüssel:** im Falle einer Beziehung mit Kardinalität
  - $M : N : P$ : Kombination aller Fremdschlüssel
  - $M : N : 1$ : Fremdschlüssel der  $M$ - und  $N$ -Relationen<sup>2</sup>Fremdschlüssel im Falle  $M : N : P$ ; im Falle  $M : N : 1$  nur die Fremdschlüssel der  $M$ - und  $N$ -Relationen
- **Attribute:** Alle einfachen Attribute bzw. einfachen Komponenten zusammengesetzter Attribute des  $M:N$  Beziehungstypen werden als Attribute zu  $R$  hinzugefügt.

---

<sup>2</sup>Generell sollte für jede 1-Seite einer  $n$ -wertigen Beziehung ein Kandidatenschlüssel erzeugt werden, der die Fremdschlüssel aller anderen Relationen der Beziehung enthält.

# Beispiel: Abbildung von n-wertigen Beziehungstypen.

- **Beispiel:** Der 3-wertige Beziehungstyp **anbieten**,



- Der Beziehungstyp wird durch eine **neue Relation** Anbieten abgebildet.
- Der **Primärschlüssel** ist die Kombination der Fremdschlüssel von Vorlesung und Semester: { VorlNr, Jahr, Semester }

Dozent[M-Nummer, ...]

Semester[Jahr, Semester, ...]

Vorlesung[VorlNr, ...]

Anbieten[VorlNr, Semester, Jahr, M-Nummer, Tag]

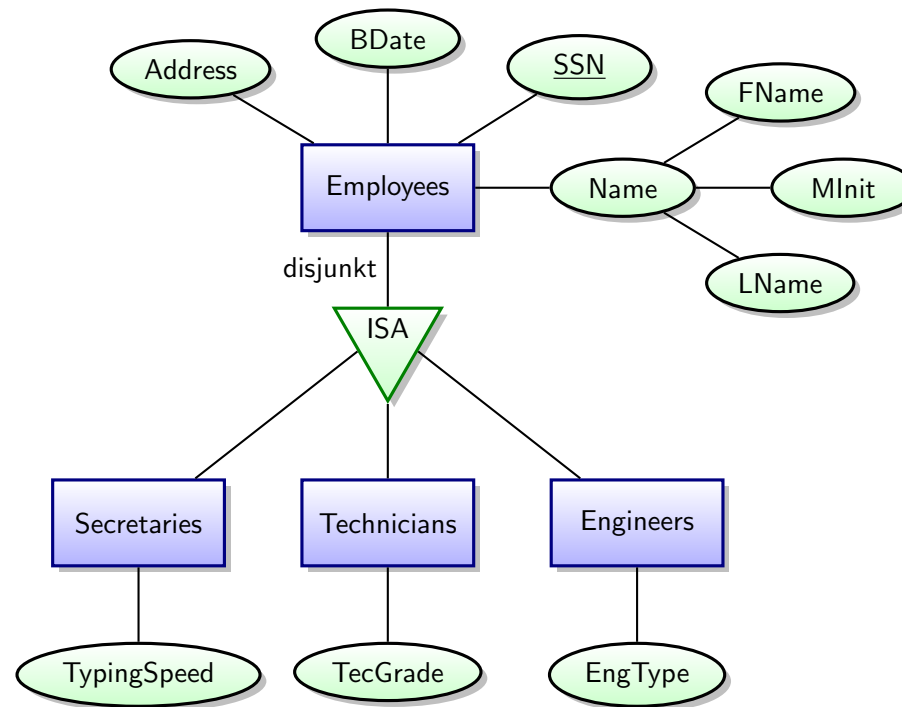
## Schritt 6: Abbildung von Spezialisierung/Generalisierung

- Notation:
  - Untertyp:  $U_1, U_2, \dots, U_m$
  - Obertyp:  $O$  mit Attributen  $k, a_1, a_2, \dots, a_n$
  - $k$  ist Primärschlüssel des Obertypen  $O$
- Umsetzung:
  - Relation  $R$  für Obertyp  $O$  mit Attributen  $attr(R) = \{\underline{k}, a_1, \dots, a_n\}$ .
  - Relation  $R_i$  für Untertypen  $U_i, 1 \leq i \leq m$ , mit den Attributen  $attr(R_i) = \{\underline{k}\} \cup \{\text{Attribute von } U_i\}$ .
  - Attribute  $k$  der Relationen  $R_i$  sind Fremdschlüssel auf Attribut  $k$  in  $R$ .
- Kann für **alle Arten der Spezialisierung** verwendet werden:
  - vollständig und partiell
  - disjunkt und überlappend
- **Einschränkung:** vollständig und/oder disjunkt wird nicht erzwungen



# Beispiel: Abbildung von Spezialisierung

- **Beispiel:** Spezialisierung von **Employees**



Employees[SSN, FName, MInit, LName, BDate, Address]

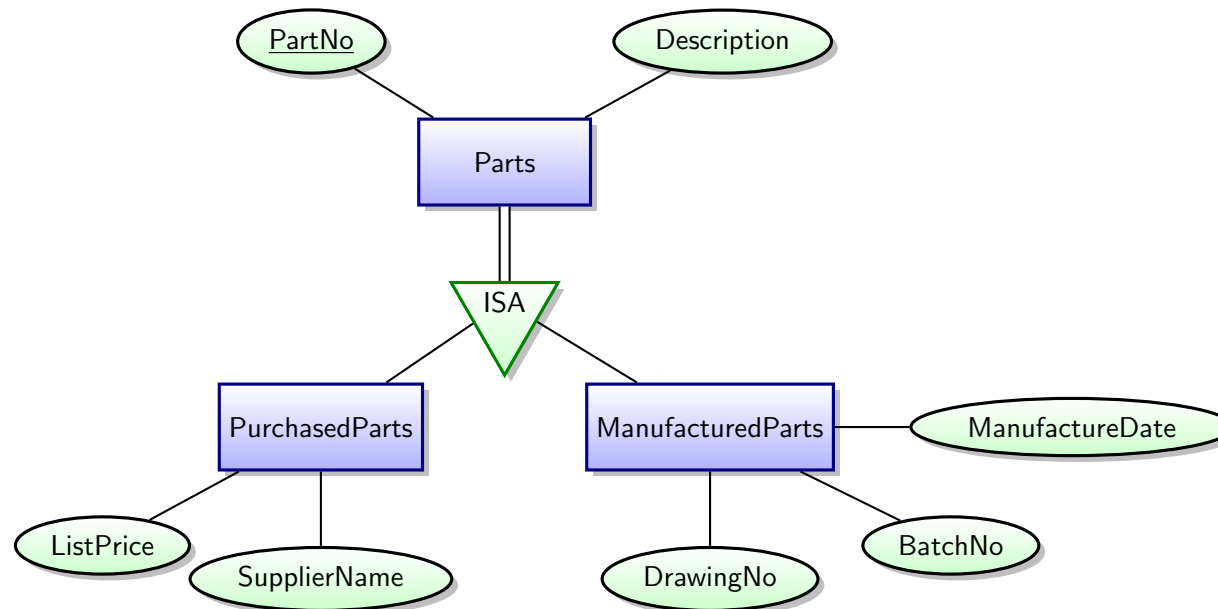
Secretaries[SSN, TypingSpeed]

Technicians[SSN, TecGrade]

Engineers[SSN, EngType]

# Integrierte Übung 3.7

- Bilden Sie folgendes ER-Diagramm in Relationen ab.



# Zusammenfassung der Abbildungen

Abbildung zwischen dem ER und dem relationalem Modell

ER Modell	Relationales Modell
Entitätstyp	Entitätsrelation
1:1 oder 1:N Beziehungstyp	Fremdschlüssel (oder Beziehungsrelation)
M:N Beziehungstyp	Beziehungsrelation mit 2 Fremdschlüsseln
$n$ -wertige Beziehungstyp	Beziehungsrelation mit $n$ Fremdschlüsseln
(Einfaches) Attribut	Attribut
zusammengesetztes Attribut	Menge von einfachen Attributen
Mehrwertiges Attribut	Relation mit Fremdschlüssel
Schlüsselattribut	Primärschlüssel
Spezialisierung	Relation für Ober- und Untertypen