

Teil IV

Datenbankentwurf

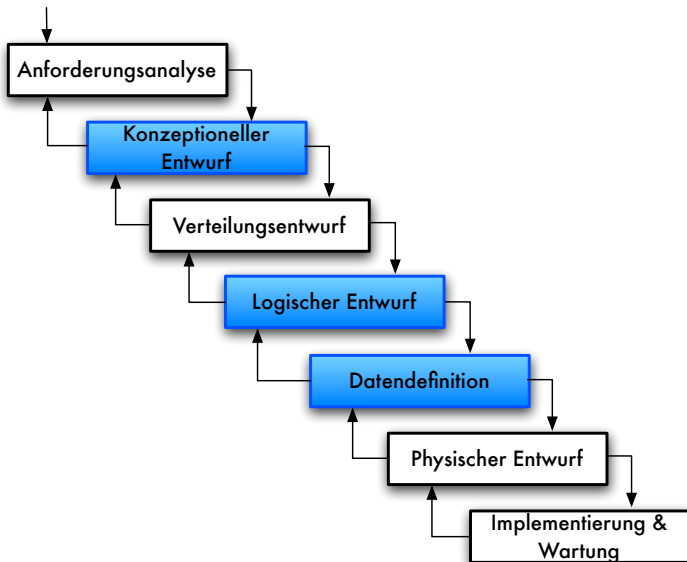
# Datenbankentwurf

- 1 Phasen des Datenbankentwurfs
- 2 Weiteres Vorgehen beim Entwurf
- 3 Kapazitätserhaltende Abbildungen
- 4 ER-auf-RM-Abbildung

# Entwurfsaufgabe

- Datenhaltung für mehrere Anwendungssysteme und mehrere Jahre
- daher: besondere Bedeutung
- Anforderungen an Entwurf
  - ▶ Anwendungsdaten jeder Anwendung sollen aus Daten der Datenbank ableitbar sein (und zwar möglichst effizient)
  - ▶ nur „vernünftige“ (wirklich benötigte) Daten sollen gespeichert werden
  - ▶ nicht-redundante Speicherung

# Phasenmodell



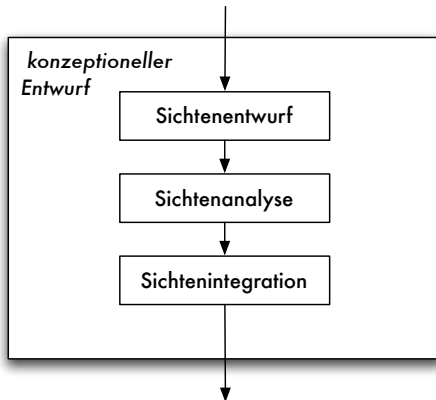
# Anforderungsanalyse

- **Vorgehensweise:** Sammlung des Informationsbedarfs in den Fachabteilungen
- **Ergebnis:**
  - ▶ informale Beschreibung (Texte, tabellarische Aufstellungen, Formblätter, usw.) des Fachproblems
  - ▶ Trennen der Information über Daten (Datenanalyse) von den Information über Funktionen (Funktionsanalyse)
- **„Klassischer“ DB-Entwurf:**
  - ▶ nur Datenanalyse und Folgeschritte
- **Funktionsentwurf:**
  - ▶ siehe Methoden des Software Engineering

# Konzeptioneller Entwurf

- erste formale Beschreibung des Fachproblems
- **Sprachmittel:** semantisches Datenmodell
- **Vorgehensweise:**
  - ▶ Modellierung von Sichten z.B. für verschiedene Fachabteilungen
  - ▶ Analyse der vorliegenden Sichten in Bezug auf Konflikte
  - ▶ Integration der Sichten in ein Gesamtschema
- **Ergebnis:** konzeptionelles Gesamtschema, z.B. ER-Diagramm

# Phasen des konzeptionellen Entwurf

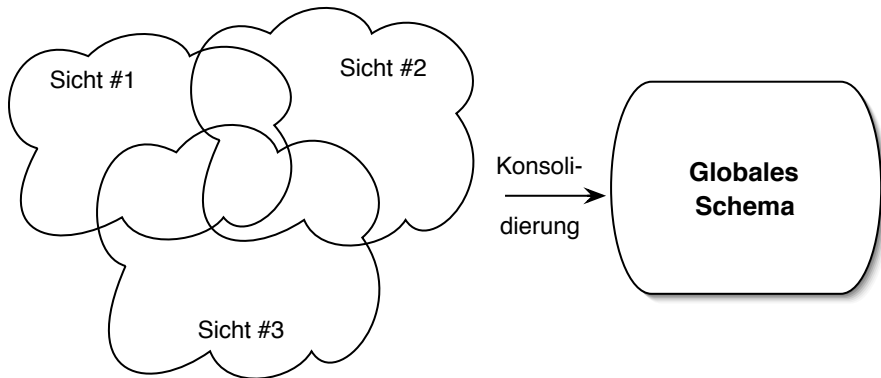


## Weiteres Vorgehen beim Entwurf

- ER-Modellierung von verschiedenen **Sichten** auf Gesamtinformation, z.B. für verschiedene Fachabteilungen eines Unternehmens  $\rightsquigarrow$  **konzeptueller Entwurf**
  - ▶ Analyse und Integration der Sichten
  - ▶ Ergebnis: konzeptionelles Gesamtschema
- Verteilungsentwurf bei verteilter Speicherung
- Abbildung auf konkretes Implementierungsmodell (z.B. Relationenmodell)  $\rightsquigarrow$  **logischer Entwurf**
- Datendefinition, Implementierung und Wartung  $\rightsquigarrow$  **physischer Entwurf**

# Sichtenintegration

- Analyse der vorliegenden Sichten in Bezug auf Konflikte
- Integration der Sichten in ein Gesamtschema



# Integrationskonflikte

- **Namenskonflikte:** Homonyme / Synonyme
  - ▶ Homonyme: Schloss; Kunde
  - ▶ Synonyme: Auto, KFZ, Fahrzeug
- **Typkonflikte:** verschiedene Strukturen für das gleiche Element
- **Wertebereichskonflikte:** verschiedene Wertebereiche für ein Element
- **Bedingungskonflikte:** z.B. verschiedene Schlüssel für ein Element
- **Strukturkonflikte:** gleicher Sachverhalt durch unterschiedliche Konstrukte ausgedrückt

## Verteilungsentwurf

- sollen Daten auf mehreren Rechnern verteilt vorliegen, muss Art und Weise der **verteilten Speicherung** festgelegt werden

- z.B. bei einer Relation

KUNDE (KNr, Name, Adresse, PLZ, Konto)

- ▶ **horizontale** Verteilung:

KUNDE\_1 (KNr, Name, Adresse, PLZ, Konto)

**where** PLZ < 50.000

KUNDE\_2 (KNr, Name, Adresse, PLZ, Konto)

**where** PLZ >= 50.000

- ▶ **vertikale** Verteilung (Verbindung über KNr Attribut):

KUNDE\_Adr (KNr, Name, Adresse, PLZ)

KUNDE\_Konto (KNr, Konto)

# Logischer Entwurf

- **Sprachmittel:** Datenmodell des ausgewählten „Realisierungs“-DBMS z.B. relationales Modell
- **Vorgehensweise:**
  - 1 (automatische) Transformation des konzeptionellen Schemas z.B. ER → relationales Modell
  - 2 Verbesserung des relationalen Schemas anhand von Gütekriterien (Normalisierung, siehe Kapitel 5):  
Entwurfsziele: Redundanzvermeidung, ...
- **Ergebnis:** logisches Schema, z.B. Sammlung von Relationenschemata

# Datendefinition

- Umsetzung des logischen Schemas in ein konkretes Schema
- **Sprachmittel:** DDL und DML eines DBMS z.B. Oracle, DB2, SQL Server
  - ▶ Datenbankdeklaration in der DDL des DBMS
  - ▶ Realisierung der Integritätssicherung
  - ▶ **Definition der Benutzersichten**

# Physischer Entwurf

- Ergänzen des physischen Entwurfs um Zugriffsunterstützung bzgl. Effizienzverbesserung, z.B. Definition von Indexen
- Index
  - ▶ Zugriffspfad: Datenstruktur für zusätzlichen, schlüsselbasierten Zugriff auf Tupel ( $\langle\langle \text{Schlüsselattributwert}, \text{Tupeladresse} \rangle\rangle$ )
  - ▶ meist als B\*-Baum realisiert
- **Sprachmittel:** *Speicherstruktursprache* SSL

# Indexe in SQL

```
create [ unique ] index indexname
  on relname (
    attrname [ asc | desc ],
    attrname [ asc | desc ],
    ...
  )
```

- Beispiel

```
create index WeinIdx on WEINE (Name)
```

## Notwendigkeit für Zugriffspfade

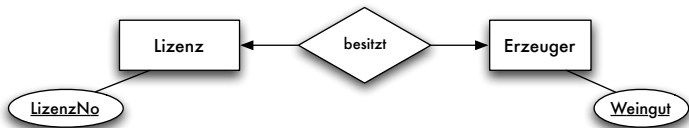
- Beispiel: Tabelle mit 100 GB Daten, Festplattentransferrate ca. 50 MB/s
- Operation: Suchen eines Tupels (Selektion)
- Implementierung: sequentielles Durchsuchen
- Aufwand:  $102.400/50 = 2.048 \text{ sec.} \approx 34 \text{ min.}$

# Implementierung und Wartung

- Phasen

- ▶ der Wartung,
- ▶ der weiteren Optimierung der physischen Ebene,
- ▶ der Anpassung an neue Anforderungen und Systemplattformen,
- ▶ der Portierung auf neue Datenbankmanagementsysteme
- ▶ etc.

# Kapazitätserhöhende Abbildung



- Abbildung auf

$$R = \{\text{LizenzNo}, \text{Weingut}\}$$

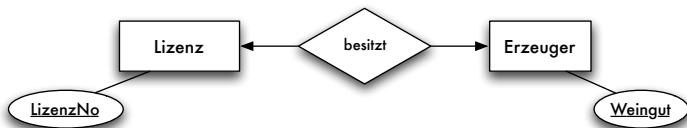
mit genau einem Schlüssel

$$K = \{\{\text{LizenzNo}\}\}$$

- mögliche ungültige Relation:

BESITZT	LizenzNo	Weingut
	007	Helena
	42	Helena

# Kapazitätserhaltende Abbildung



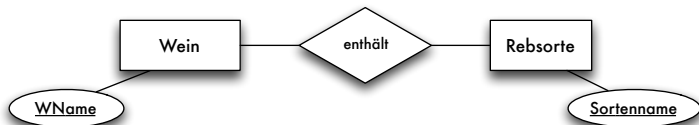
- korrekte Ausprägung

BESITZT	LizenzNo	Weingut
	007	Helena
	42	Müller

- korrekte Schlüsselmenge

$$K = \{\{LizenzNo\}, \{Weingut\}\}$$

# Kapazitätsvermindernde Abbildung



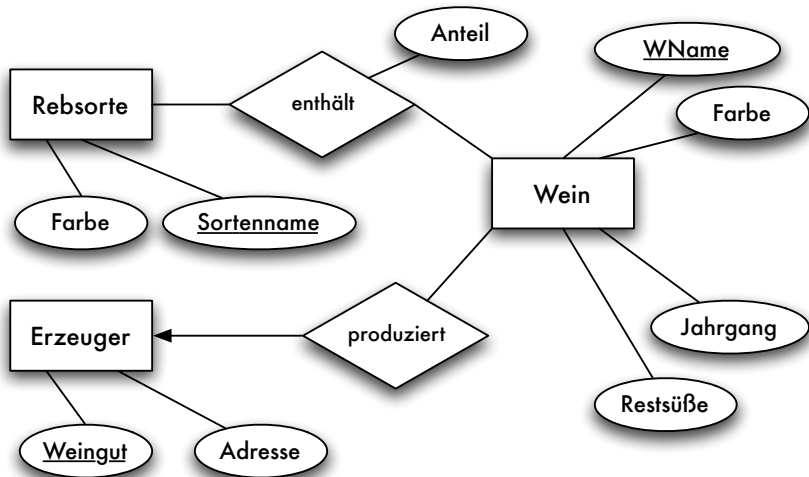
- Relationenschema mit einem Schlüssel {WName}
- als Ausprägung nicht mehr möglich:

ENTHÄLT	WName	Sortenname
	Zinfandel Red Blossom	Zinfandel
	Bordeaux Blanc	Cabernet Sauvignon
	Bordeaux Blanc	Muscadelle

- kapazitätserhaltend mit Schlüssel beider Entity-Typen im Relationenschema als neuer Schlüssel

$$K = \{\{WName, Sortenname\}\}$$

# Beispielabbildung ER-RM: Eingabe



## Beispielabbildung ER-RM: Ergebnis

- 1 REBSORTE = {Farbe, Sortenname} mit  $K_{\text{REBSORTE}} = \{\{\text{Sortenname}\}\}$
- 2 ENTHÄLT = {Sortenname, WName, Anteil} mit  $K_{\text{ENTHÄLT}} = \{\{\text{Sortenname}, \text{WName}\}\}$
- 3 WEIN = {Farbe, WName, Jahrgang, Restsüße} mit  $K_{\text{WEIN}} = \{\{\text{WName}\}\}$
- 4 PRODUZIERT = {WName, Weingut} mit  $K_{\text{PRODUZIERT}} = \{\{\text{WName}\}\}$
- 5 ERZEUGER = {Weingut, Adresse} mit  $K_{\text{ERZEUGER}} = \{\{\text{Weingut}\}\}$

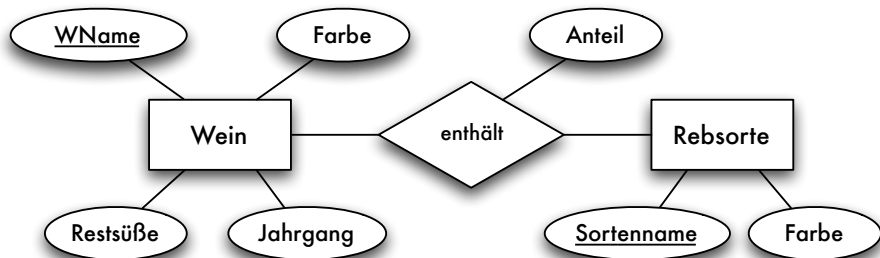
# ER-Abbildung auf Relationen

- **Entity-Typen und Beziehungstypen**: jeweils auf Relationenschemata
- **Attribute**: Attribute des Relationenschemas, **Schlüssel** werden übernommen
- **Kardinalitäten** der Beziehungen: durch Wahl der Schlüssel bei den zugehörigen Relationenschemata ausgedrückt
- in einigen Fällen: **Verschmelzen** der Relationenschemata von Entity- und Beziehungstypen
- zwischen den verbleibenden Relationenschemata diverse Fremdschlüsselbedingungen einführen

# Abbildung von Beziehungstypen

- neues Relationenschema mit allen Attributen des Beziehungstyps, zusätzlich Übernahme aller Primärschlüssel der beteiligten Entity-Typen
- **Festlegung der Schlüssel:**
  - ▶ **m:n-Beziehung:** beide Primärschlüssel zusammen werden Schlüssel im neuen Relationenschema
  - ▶ **1:n-Beziehung:** Primärschlüssel der n-Seite (bei der funktionalen Notation die Seite ohne Pfeilspitze) wird Schlüssel im neuen Relationenschema
  - ▶ **1:1-Beziehung:** beide Primärschlüssel werden je ein Schlüssel im neuen Relationenschema, der Primärschlüssel wird dann aus diesen Schlüsseln gewählt

# n:m-Beziehungen

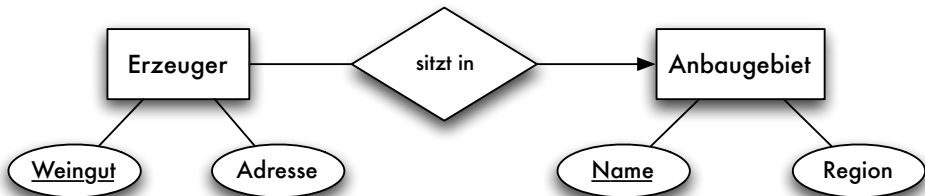


## • Umsetzung

- 1 REBSORTE = {Farbe, Sortenname} mit  $K_{\text{REBSORTE}} = \{\{\text{Sortenname}\}\}$
- 2 ENTHÄLT = {Sortenname, WName, Anteil} mit  $K_{\text{ENTHÄLT}} = \{\{\text{Sortenname}, \text{WName}\}\}$
- 3 WEIN = {Farbe, WName, Jahrgang, Restsüße} mit  $K_{\text{WEIN}} = \{\{\text{WName}\}\}$

## • Attribute Sortenname und WName sind gemeinsam Schlüssel

# 1:n-Beziehungen



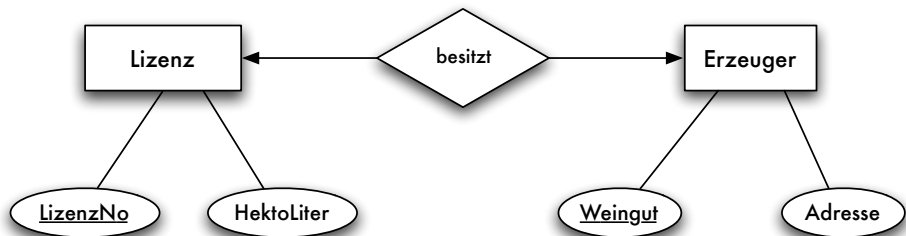
- Umsetzung (zunächst)

- ▶ ERZEUGER mit den Attributen Weingut und Adresse,
- ▶ ANBAUGEBIET mit den Attributen Name und Region und
- ▶ SITZT\_IN mit den Attributen Weingut und Name und dem Primärschlüssel der *n*-Seite Weingut als Primärschlüssel dieses Schemas.

# Mögliche Verschmelzungen

- **optionale Beziehungen** ( $[0,1]$  oder  $[0,n]$ ) werden nicht verschmolzen
- bei Kardinalitäten  $[1,1]$  oder  $[1,n]$  (**zwingende Beziehungen**) Verschmelzung möglich:
  - ▶ **1:n-Beziehung**: das Entity-Relationenschema der n-Seite kann in das Relationenschema der Beziehung integriert werden
  - ▶ **1:1-Beziehung**: beide Entity-Relationenschemata können in das Relationenschema der Beziehung integriert werden

# 1:1-Beziehungen



- Umsetzung (zunächst)

- ▶ ERZEUGER mit den Attributen Weingut und Adresse
- ▶ LIZENZ mit den beiden Attributen LizenzNo und Hektoliter
- ▶ BESITZT mit den Primärschlüsseln der beiden beteiligten Entity-Typen jeweils als Schlüssel dieses Schemas, also LizenzNo und Weingut

# 1:1-Beziehungen: Verschmelzung

- Umsetzung mit Verschmelzung

- ▶ verschmolzene Relation:

ERZEUGER	Weingut	Adresse	LizenzNo	Hektoliter
	Rotkäppchen	Freiberg	42-007	10.000
	Weingut Müller	Dagstuhl	42-009	250

- ▶ Erzeuger ohne Lizenz erfordern Nullwerte:

ERZEUGER	Weingut	Adresse	LizenzNo	Hektoliter
	Rotkäppchen	Freiberg	42-007	10.000
	Weingut Müller	Dagstuhl	⊥	⊥

- ▶ freie Lizenzen führen zu weiteren Nullwerten:

ERZEUGER	Weingut	Adresse	LizenzNo	Hektoliter
	Rotkäppchen	Freiberg	42-007	10.000
	Weingut Müller	Dagstuhl	⊥	⊥
	⊥	⊥	42-003	100.000

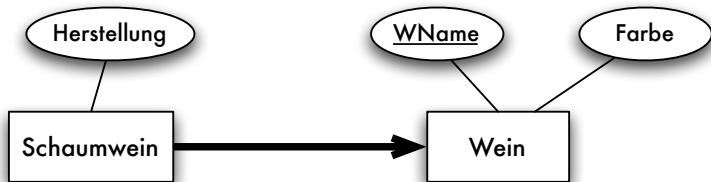
# Abhängige Entity-Typen



## • Umsetzung

- 1 WEINJAHRGANG = {WName, Restsüße, Jahr} mit  $K_{\text{WEINJAHRGANG}} = \{\{WName, Jahr\}\}$
- 2 WEIN = {Farbe, WName} mit  $K_{\text{WEIN}} = \{\{WName\}\}$ 
  - ▶ Attribut WName in WEINJAHRGANG ist Fremdschlüssel zur Relation WEIN

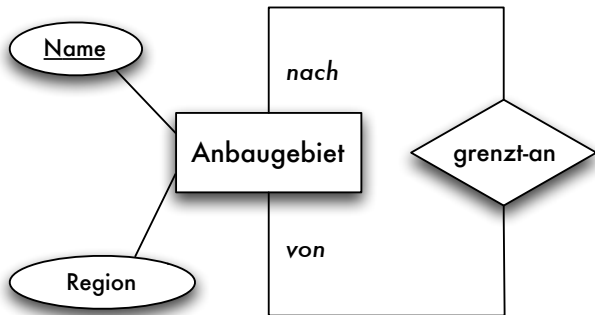
# IST-Beziehung



## • Umsetzung

- 1 WEIN = {Farbe, WName, Jahrgang, Restsüße} mit  $K_{\text{WEIN}} = \{\{WName\}\}$
  - 2 SCHAUMWEIN = {WName, Herstellung} mit  $K_{\text{SCHAUMWEIN}} = \{\{WName\}\}$
- ▶ WName in SCHAUMWEIN ist Fremdschlüssel bezüglich der Relation WEIN

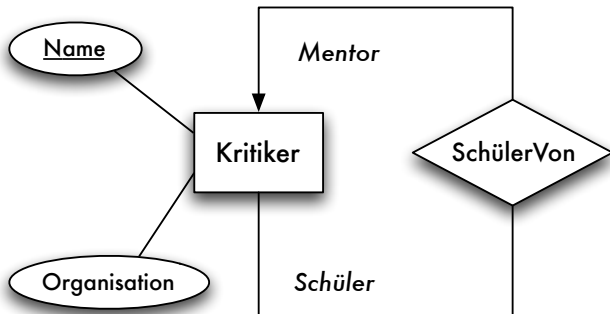
# Rekursive Beziehungen



- Umsetzung

- 1 ANBAUGEBIET = {Name, Region} mit  $K_{\text{ANBAUGEBIET}} = \{\{\text{Name}\}\}$
- 2 GRENZT\_AN = {nach, von} mit  $K_{\text{GRENZT\_AN}} = \{\{\text{nach, von}\}\}$

# Rekursive funktionale Beziehungen

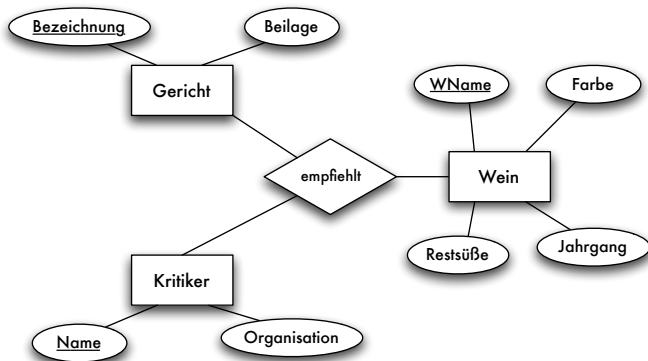


## • Umsetzung

①  $KRITIKER = \{Name, Organisation, Mentorname\}$  mit  
 $K_{KRITIKER} = \{\{Name\}\}$

- ▶ **Mentorname** ist Fremdschlüssel auf das Attribut **Name** der Relation **KRITIKER**.

# Mehrstellige Beziehungen



- jeder beteiligte Entity-Typ wird nach den obigen Regeln behandelt
- für Beziehung `empfehlt` werden Primärschlüssel der drei beteiligten Entity-Typen in das resultierende Relationenschema aufgenommen
- Beziehung ist allgemeiner Art (k:m:n-Beziehung): alle Primärschlüssel bilden zusammen den Schlüssel

## Mehrstellige Beziehungen: Ergebnis

- 1 EMPFIEHLT = {WName, Bezeichnung, Name} mit  $K_{\text{EMPFIEHLT}} = \{\{WName, Bezeichnung, Name\}\}$
  - 2 GERICHT = {Bezeichnung, Beilage} mit  $K_{\text{GERICHT}} = \{\{Bezeichnung\}\}$
  - 3 WEIN = {Farbe, WName, Jahrgang, Restsüße} mit  $K_{\text{WEIN}} = \{\{WName\}\}$
  - 4 KRITIKER = {Name, Organisation} mit  $K_{\text{KRITIKER}} = \{\{Name\}\}$
- Die drei Schlüsselattribute von EMPFIEHLT sind Fremdschlüssel für die jeweiligen Ursprungsrelationen.

# Übersicht über die Transformationen

ER-Konzept	wird abgebildet auf relationales Konzept
Entity-Typ $E_i$ Attribute von $E_i$ Primärschlüssel $P_i$	Relationenschema $R_i$ Attribute von $R_i$ Primärschlüssel $P_i$
Beziehungstyp  dessen Attribute $1 : n$ $1 : 1$ $m : n$	Relationenschema Attribute: $P_1, P_2$ weitere Attribute $P_2$ wird Primärschlüssel der Beziehung $P_1$ und $P_2$ werden Schlüssel der Beziehung $P_1 \cup P_2$ wird Primärschlüssel der Beziehung
IST-Beziehung	$R_1$ erhält zusätzlichen Schlüssel $P_2$

$E_1, E_2$ : an Beziehung beteiligte Entity-Typen,

$P_1, P_2$ : deren Primärschlüssel,

$1 : n$ -Beziehung:  $E_2$  ist  $n$ -Seite,

IST-Beziehung:  $E_1$  ist speziellerer Entity-Typ

# Zusammenfassung

- Phasen des Datenbankentwurfs
- Datenbankmodell, Datenbankschema, Datenbank(instanz)
- Entity-Relationship-Modell
- ER-Erweiterungen: Spezialisierung, Generalisierung, Partitionierung
- weitere Entwurfsschritte