# Syntax im Relationenmodell

| Konzept | Heuer/Saake 2000 | Kemper/Eickler 2004 | Vossen 2000 | Stein 2004-2015 |
|---|---|---|---|---|
| Attribut | $A, A_i, B$ | $A, A_i, B$ | $A, A_i, B$ | $A, A_i, B$ |
| Domäne von Attribut | $dom(A_i), D_i$ | $dom(A_i), D_i$ | $dom(A_i)$ | $dom(A_i)$ |
| Attributmenge | $X, Y$ | $\alpha, \beta$ | $X$ | $\alpha, \beta, \{A_{i_1}, ..., A_{i_k}\}$ |
| Domäne von Attributmenge | $dom(X)$ | – | – | – |
| Relationenschema | $R$ | $\mathcal{R}, \mathcal{S}$ | $X$ | $\mathcal{R}$ |
| Tupel | $t : R \to \bigcup_{D_i}$ | $r, s$ | $\mu : X \to dom(X)$ | $t : \mathcal{R} \to \bigcup dom(A_i)$ |
| Menge aller Tupel über Attributmenge | – | – | $\mathsf{Tup}(X)$ | – |
| Teiltupel | $t(X)$ | $r.\alpha, s.\kappa$ | – | $t(\alpha)$ |
| Relation | $r(R), r$ | $R, S$ | $r$ | $r(\mathcal{R}), r$ |
| Menge aller Relationen über Schema | $\mathbf{REL}(R) = \{r \mid r(R)\}$ | – | $\mathsf{Rel}(X) = \{r \mid r \subseteq \mathsf{Tup}(X)\}$ | $\{r \mid r(\mathcal{R})\}$ |
| Datenbankschema | $S = \{R_1, ..., R_p\}$ | – | $\mathbf{R} = \{R_1, ..., R_k\}$ | $\boldsymbol{\mathcal{R}} = \{\mathcal{R}_1, ..., \mathcal{R}_p\}$ |
| Datenbank | $d(S) = \{r_1, ..., r_p\}, r_i \in \mathbf{REL}(R)$ | – | $d(\mathbf{R}) = \{r_1, ..., r_k\}, r_i \in \mathsf{Rel}(X)$ | $d(\boldsymbol{\mathcal{R}}) = \{r_1, ..., r_p\}$ |
| Menge aller punktweise konsistenten Datenbanken | – | – | $\mathsf{Dat}(\mathbf{R})$ | – |
| Menge aller Datenbanken | – | – | $\mathsf{Sat}(\mathbf{R})$ | – |
| lokale Integritätsbedingung / intrarelationale Abhängigkeit | $b : \{r \mid r(R)\} \to \{\mathsf{true}, \mathsf{false}\}$ | – | $\sigma : \mathsf{Rel}(X) \to \{0, 1\}$ | $b : \{r \mid r(\mathcal{R})\} \to \{\mathsf{true}, \mathsf{false}\}$ |
| Menge lokaler Integritätsbedingungen | $\mathcal{B}$ | – | $\Sigma_X$ | – |
| globale Integritätsbedingung / interrelationale Abhängigkeit | $\gamma : \{d \mid d(S)\} \to \{\mathsf{true}, \mathsf{false}\}$ | – | $\sigma : \mathsf{Dat}(\mathbf{R}) \to \{0, 1\}$ | $b : \{d \mid d(\boldsymbol{\mathcal{R}})\} \to \{\mathsf{true}, \mathsf{false}\}$ |
| Menge globaler Integritätsbedingungen | $\Gamma$ | – | $\Sigma_{\mathbf{R}}$ | – |
| erweitertes Relationenschema | $\mathcal{R} = (R, \mathcal{B})$ | – | $R = (X, \Sigma_X)$ | – |
| Menge aller Relationen, die lokale Integritätsbedingungen erfüllen | $\mathbf{SAT}_R(\mathcal{B}) = \{r \mid r(\mathcal{R})\}$ | – | $\mathsf{Sat}(X, \Sigma_X)$ | – |
| lokal erweitertes Datenbankschema | $S = \{\mathcal{R}_1, ..., \mathcal{R}_p\}$ | – | – | – |
| global erweitertes Datenbankschema | $\mathcal{S} = (S, \Gamma)$ | – | $\mathbf{D} = (\mathbf{R}, \Sigma_{\mathbf{R}})$ | – |
| Menge aller Datenbanken, die globale Integritätsbedingungen erfüllen | $\mathbf{SAT}(\mathcal{S}) = \{d \mid d(\mathcal{S})\}$ | – | $\mathsf{Sat}(\Sigma_{\mathbf{R}})$ | – |
| Schlüssel | $K$ | $\kappa$ | $K$ | $\kappa$ |
| Fremdschlüsselbedingung | $X(R_1) \to Y(R_2)$ | – | – | – |
| Funktionale Abhängigkeit | $X \to Y$ | $\alpha \to \beta$ | $X \to Y$ | $\alpha \to \beta$ |
| volle funktionale Abhängigkeit | – | $\alpha \xrightarrow{\bullet} \beta$ | – | – |
| Menge funktionaler Abhängigkeiten | $F$ | $F$ | $F$ | $F$ |
| Hülle funktionaler Abhängigkeiten | $F^+$ | $F^+$ | $F^+$ | $F^+$ |

# Kapitel DB:IV

## IV. Logischer Datenbankentwurf mit dem relationalen Modell

- Das relationale Modell
- Integritätsbedingungen
- Umsetzung ER-Schema in relationales Schema

# Das relationale Modell

- Im relationalen Modell werden die Objekttypen der zu modellierenden Anwendungswelt durch *Relationenschemata* beschrieben.
- Ein Relationenschema besteht aus einer Menge von *Attributen*, die Eigenschaften der Objekte eines Objekttyps repräsentieren.
- Attribute haben als *Wertebereiche (Domains)* einfache Mengen, die meist von einem Standard-Datentyp wie `Integer`, `String` oder `Boolean` sind.
- Legt man eine Reihenfolge der Attribute fest, so kann eine *Relation* als Teilmenge des kartesischen Produktes über die Wertebereiche der Attribute eines Relationenschemas aufgefasst werden.
  Eine Relation wird auch als *Instanz* eines Relationenschemas bezeichnet.
- Ein *Datenbankschema* besteht aus einer Menge von Relationenschemata.
- Die Menge der zu einem bestimmten Zeitpunkt zu den Relationenschemata eines Datenbankschemas vorhandenen Relationen heißt *Datenbank* oder *Datenbankzustand*.

# Das relationale Modell

Interpretation einer Relation als Tabelle:

| Buch | | | |
|---|---|---|---|
| Inv_Nr | Titel | ISBN | Autor |
| 0110 | Lesebuch | 2-341... | Popper |
| 1201 | C++ | 2-123... | Stroustrup |
| 3309 | Längengrad | 2-123... | Sobel |
| 4711 | Glücksformel | 2-679... | Klein |
| 7510 | Heuristics | 9-212... | Pearl |

} Relationenname

} Relationenschema

} Relation = Instanz des Relationenschemas

Bemerkungen:

- Andere Bezeichnungen für den logischen Datenbankentwurf:
  logischer Entwurf, Implementationsentwurf
- Die Begriffe „relationales Modell“ und „Relationenmodell“ werden synonym verwendet.
- Die Tabellensicht unterstellt eine (beliebige aber feste) Ordnung der Attribute; das Relationenmodell macht das nicht.

# Das relationale Modell

**Definition 1 (Relationenschema, Tupel, Relation)**

Sei $\mathcal{R} = \{A_1, \ldots, A_n\}$ eine endliche Menge von Attributen mit nichtleeren Wertebereichen $dom(A_1), \ldots, dom(A_n)$. Dann wird $\mathcal{R}$ als Relationenschema bezeichnet; ein $v \in dom(A_i)$ wird Attributwert für $A_i$ genannt.

# Das relationale Modell

**Definition 1 (Relationenschema, Tupel, Relation)**

Sei $\mathcal{R} = \{A_1, \ldots, A_n\}$ eine endliche Menge von Attributen mit nichtleeren Wertebereichen $dom(A_1), \ldots, dom(A_n)$. Dann wird $\mathcal{R}$ als Relationenschema bezeichnet; ein $v \in dom(A_i)$ wird Attributwert für $A_i$ genannt.

1. Jedes (einzelne) Tupel über $\mathcal{R}$ ist eine totale Abbildung

$$t : \mathcal{R} \to \bigcup_{i=1}^{n} dom(A_i),$$

mit $t(A) \in dom(A)$ für alle $A \in \mathcal{R}$.

# Das relationale Modell

## Definition 1 (Relationenschema, Tupel, Relation)

Sei $\mathcal{R} = \{A_1, \ldots, A_n\}$ eine endliche Menge von Attributen mit nichtleeren Wertebereichen $dom(A_1), \ldots, dom(A_n)$. Dann wird $\mathcal{R}$ als Relationenschema bezeichnet; ein $v \in dom(A_i)$ wird Attributwert für $A_i$ genannt.

1. Jedes (einzelne) Tupel über $\mathcal{R}$ ist eine totale Abbildung

$$t : \mathcal{R} \to \bigcup_{i=1}^{n} dom(A_i),$$

   mit $t(A) \in dom(A)$ für alle $A \in \mathcal{R}$.

2. Für eine Teilmenge der Attribute, $\alpha \subseteq \mathcal{R}$, bezeichne $t|_\alpha$ die Einschränkung der Abbildung $t$ auf die Menge $\alpha$, also ein Teiltupel über $\mathcal{R}$.
   In Anlehnung an die Datenbankliteratur notieren wir die Abbildung bzw. das Teiltupel $t|_\alpha$ vereinfachend auch als $t(\alpha)$.

3. Eine Relation $r$ über $\mathcal{R}$, in Zeichen auch $r(\mathcal{R})$, ist eine endliche Menge von Tupeln über $\mathcal{R}$.

Bemerkungen:

- Jedes Objekt der Anwendungswelt wird hier durch eine eigene Tupelfunktion $t$ repräsentiert. Ein Objekt lässt sich als *eine* Zeile in einer Tabelle in der entsprechenden Relation auffassen, wobei die Spalten der Tabelle in beliebiger Reihenfolge angegeben sein dürfen:

$$t = \{(A_1, t(A_1)),\ (A_2, t(A_2)),\ \ldots, (A_n, t(A_n))\}$$

- Vergleiche hierzu die Mathematik; hier wird ein Tupel als eine geordnete Folge fester Länge von Werten bezeichnet:

$$t \in \mathit{dom}(A_1) \times \mathit{dom}(A_2) \times \ldots \times \mathit{dom}(A_n)$$

- Im Entity-Relationship-Modell wird ein Attribut $A$ als eine Funktion betrachtet, die jeder Instanz eines Entity-Typs eine Eigenschaftsausprägung zuordnet; der Datentyp eines Attributs $A$ wird mit $T$ bezeichnet, in Zeichen: $A : T$. [DB:III ER-Konzepte und ihre Semantik]
  Im Relationenmodell wird ein Attribut $A$ lediglich als Bezeichner eines Datentyps verstanden. Somit entspricht der Wertebereich $\mathit{dom}(A)$ eines Attributes $A$ im Relationenmodell der Menge $\mathit{dom}(T)$ im Entity-Relationship-Modell.

- Zur Bezeichnung einer Relation können $r$ und $r(\mathcal{R})$ gleichermaßen verwendet werden – abhängig davon, ob auf das Relationenschema $\mathcal{R}$ im aktuellen Kontext Bezug genommen werden muss.

# Das relationale Modell

Beispiel:

- $\mathcal{R}$ = Pers_Telefon = {PANr, Telefon}

  $dom(\text{PANr}) = \texttt{Integer}$

  $dom(\text{Telefon}) = \texttt{String}$

  $r(\mathcal{R}) = r(\text{Pers\_Telefon}) = \{t_1, t_2, t_3, \ldots\}$

| Pers_Telefon | |
|---|---|
| PANr | Telefon |
| 4711 | 038203-12230 |
| 4711 | 0381-498-3427 |
| 5588 | 0391-34677 |
| ... | |

# Das relationale Modell

Beispiel:

- $\mathcal{R}$ = Pers_Telefon = {PANr, Telefon}

  $dom(\text{PANr}) = \texttt{Integer}$
  $dom(\text{Telefon}) = \texttt{String}$

  $r(\mathcal{R}) = r(\text{Pers\_Telefon}) = \{t_1, t_2, t_3, \ldots\}$

| Pers_Telefon | |
|---|---|
| PANr | Telefon |
| 4711 | 038203-12230 |
| 4711 | 0381-498-3427 |
| 5588 | 0391-34677 |
| ... | |

- $t$ : {PANr, Telefon} $\rightarrow \texttt{Integer} \cup \texttt{String}$

  $t_1 = \{(\text{PANr}, 4711), (\text{Telefon}, \text{038203-12230})\}$
  $t_1(\text{PANr}) = 4711$
  $t_1(\text{Telefon}) = \text{038203-12230}$

  $t_2 = \{(\text{PANr}, 4711), (\text{Telefon}, \text{0381-498-3427})\}$
  $t_2(\text{PANr}) = 4711$
  $t_2(\text{Telefon}) = \text{0381-498-3427}$

Bemerkungen:

- Eine Relation ist als endliche Menge von Abbildungen $t_i$ definiert. Attributwerte können über den Attributnamen eindeutig identifiziert werden.
- Fasst man jedoch eine Relation als Teilmenge eines kartesischen Produktes auf, $r \subseteq \mathit{dom}(\text{PANr}) \times \mathit{dom}(\text{Telefon})$, so werden Attributwerte durch ihre Position identifiziert.

  Insbesondere ist $\mathit{dom}(\text{PANr}) \times \mathit{dom}(\text{Telefon}) \neq \mathit{dom}(\text{Telefon}) \times \mathit{dom}(\text{PANr})$.

# Das relationale Modell

## Definition 2 (Datenbankschema, Datenbankzustand, Basisrelation)

1. Eine Menge von Relationenschemata $\boldsymbol{\mathcal{R}} = \{\mathcal{R}_1, \ldots, \mathcal{R}_p\}$ heißt Datenbankschema.

2. Eine Datenbank bzw. ein Datenbankzustand $d$ über einem Datenbankschema $\boldsymbol{\mathcal{R}} = \{\mathcal{R}_1, \ldots, \mathcal{R}_p\}$ ist eine Menge von Relationen:

$$d(\boldsymbol{\mathcal{R}}) = \{r_1, \ldots, r_p\}, \quad \text{mit } r_i = r_i(\mathcal{R}_i),\ i = 1, \ldots, p$$

3. Eine Relation $r \in d(\boldsymbol{\mathcal{R}})$ heißt Basisrelation.

Bemerkungen:

- Zur Bezeichnung einer Datenbank können $d$ und $d(\mathcal{R})$ gleichermaßen verwendet werden – abhängig davon, ob auf das Datenbankschema $\mathcal{R}$ im aktuellen Kontext Bezug genommen werden muss.
- Eine Datenbank $d$ hat zu einem Zeitpunkt $i$ den Zustand $state_i$.
  [DB:II Datenbankmodelle » Datenbankzustand]
- Der Begriff der Basisrelation dient als Unterscheidung zu den hieraus *abgeleiteten Relationen*, die nicht in der Datenbank gespeichert sind.

# Das relationale Modell

Beispiel:

- Datenbankschema $\mathcal{R} = \{$ Personen, Pers_Telefon $\}$
- Datenbankzustand $d(\mathcal{R}) = \{r_1(\text{Personen}),\ r_2(\text{Pers\_Telefon})\}$

$r_1$

| Personen | | | | |
|---|---|---|---|---|
| PANr | Vorname | Nachname | PLZ | ORT |
| 4711 | Andreas | Heuer | 18209 | DBR |
| 5588 | Gunter | Saake | 39106 | MD |
| 6834 | Michael | Korn | 39104 | MD |
| ... | | | | |

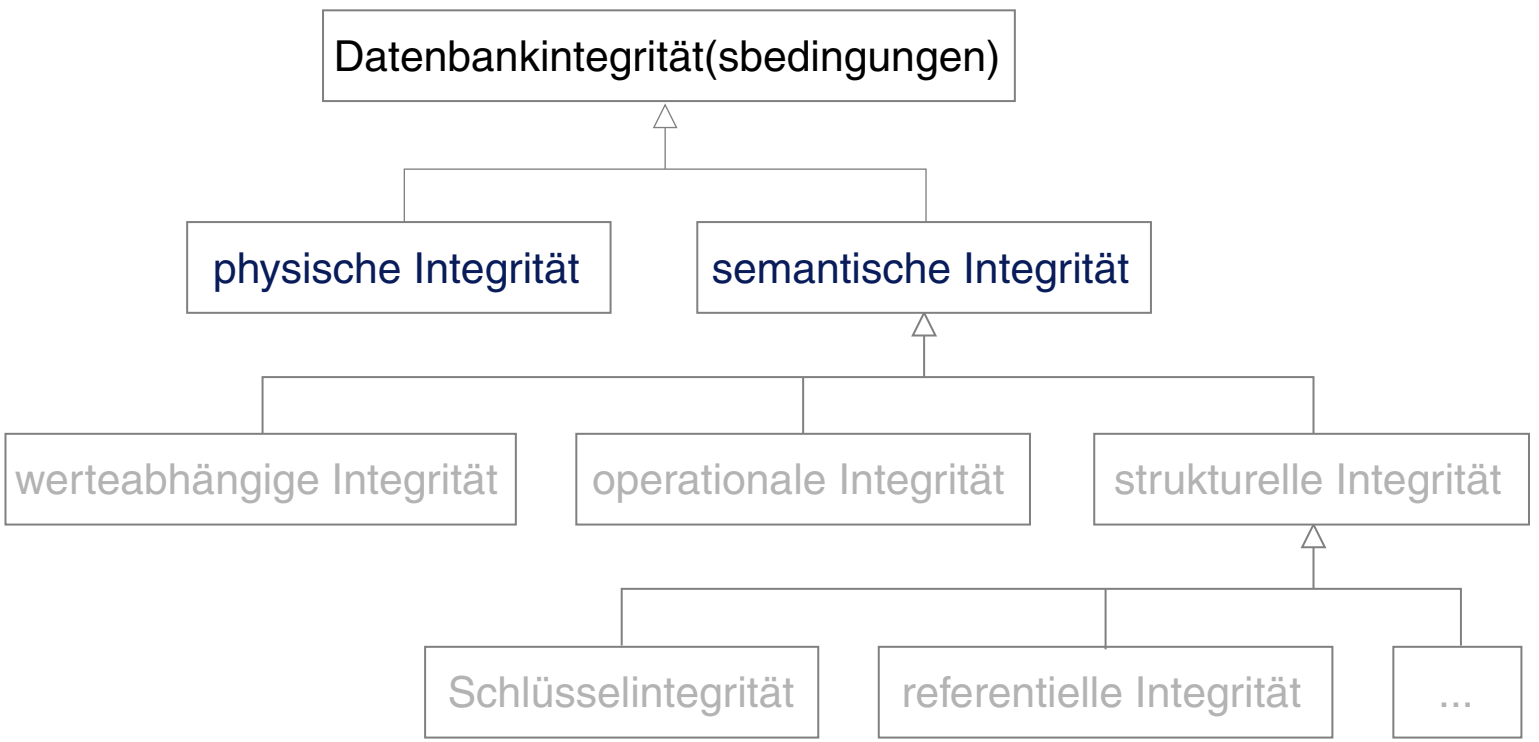
$r_2$

| Pers_Telefon | |
|---|---|
| PANr | Telefon |
| 4711 | 038203-12230 |
| 4711 | 0381-498-3427 |
| 5588 | 0391-34677 |
| ... | |

# Integritätsbedingungen

## Definition 3 (Integrität)

Die Integrität einer Datenbank $d$ bezeichnet die korrekte und widerspruchsfreie Speicherung von Daten in $d$.




[Weber 2003]

Bemerkungen:

- Zur Einhaltung bzw. Gewährleistung der Integrität werden Integritätsbedingungen formuliert.
- Die „Reichweite“ einer Integritätsbedingung, also die Anzahl der betroffenen Relationen, sollte möglichst gering gehalten werden.
- Art und Umfang der realisierten Integritätskonzepte variieren deutlich in kommerziellen Datenbankprodukten.
- Ein flexibler Mechanismus zur Sicherung komplexer Integritätsbedingungen sind Trigger. Ein Trigger ist eine benutzerdefinierte Funktion, die automatisch aufgerufen wird, sobald eine bestimmte Bedingung erfüllt ist. Einsatz für Trigger sind z.B. Tabellen mit Statistiken, in denen Werte abgeleiteter Attribute bzw. Spalten zu berechnen sind.

# Integritätsbedingungen

## Physische Integrität

- Regelt alle Aspekte der physikalischen Datenspeicherung.
- Berücksichtigt Hard- und Softwarefehler (soweit möglich).
- Behandelt Probleme des gleichzeitigen Zugriffs auf Daten.

# Integritätsbedingungen

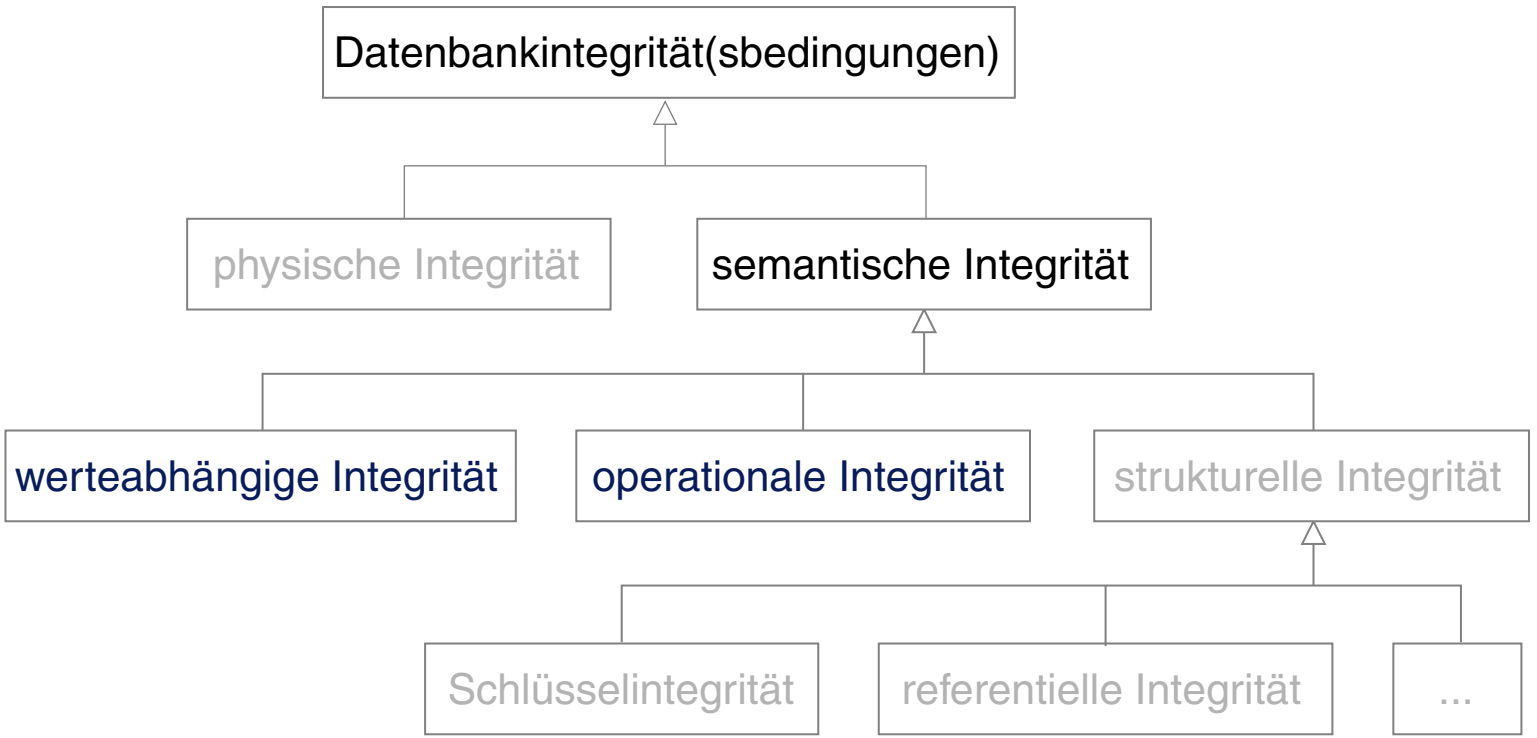
## Physische Integrität

- Regelt alle Aspekte der physikalischen Datenspeicherung.
- Berücksichtigt Hard- und Softwarefehler (soweit möglich).
- Behandelt Probleme des gleichzeitigen Zugriffs auf Daten.

## Semantische Integrität

- Wird aus den Eigenschaften des zu modellierenden Weltausschnitts abgeleitet und *definiert die zulässigen Zustände der Datenbank*:
  Legt fest, welche Werte erlaubt und welche Beziehungen zwischen Datenelementen möglich sind. [DB:III Charakterisierung von Beziehungstypen]
- Das Datenbank-Management-System überprüft mit semantischen Integritätsbedingungen, ob eine gewünschte Änderung der Datenbank zulässig ist. Das betrifft die Ebene 2 im DBMS. [DB:I DBMS » Komponenten]
- Semantische Integritätsbedingungen werden nach dem Inhalt unterschieden, den sie behandeln – in werteabhängige, operationale und strukturelle Integritätsbedingungen.

# Integritätsbedingungen




[Weber 2003]

# Integritätsbedingungen

## Werteabhängige (= statische) Integrität

Werteabhängige Integrität bezeichnet die Forderung, nicht alle durch Entity-Typen und Beziehungs-Typen definierten Datenbankzustände zuzulassen:

- Werteabhängige Integritätsbedingungen treffen Aussagen über den Zusammenhang von Werten zwischen Datenelementen.
- Sie beschränken die in der Anwendungsoberfläche evtl. möglichen Datenelemente auf Teilmengen.

# Integritätsbedingungen

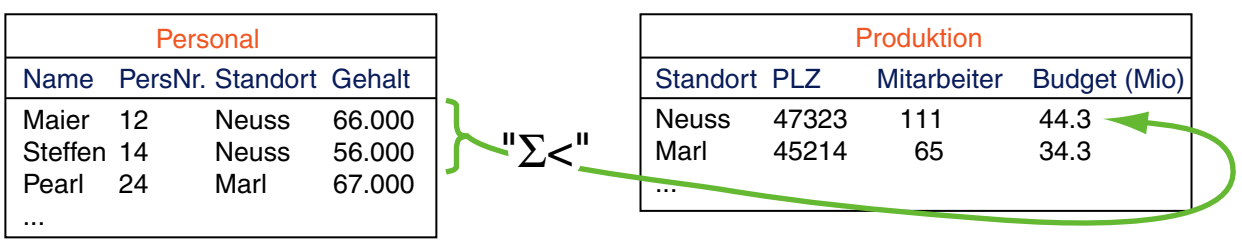
## Werteabhängige (= statische) Integrität

Werteabhängige Integrität bezeichnet die Forderung, nicht alle durch Entity-Typen und Beziehungs-Typen definierten Datenbankzustände zuzulassen:

- Werteabhängige Integritätsbedingungen treffen Aussagen über den Zusammenhang von Werten zwischen Datenelementen.
- Sie beschränken die in der Anwendungsoberfläche evtl. möglichen Datenelemente auf Teilmengen.

Beispiele:

- *Lokal*, innerhalb eines Tupels oder einer Relation: Die PLZ muss zwischen 1000 und 99999 liegen.
- *Global*, zwischen Relationen: die Summe der Gehälter aller Mitarbeiter an einem Standort darf dessen Personalbudget nicht überschreiten.



# Integritätsbedingungen

## Operationale (= dynamische) Integrität

Operationale Integritätsbedingungen gewährleisten korrekte Zustandsübergänge:

- Sie überwachen die Ausführung von Operationen.
- Durch die Angabe von Integritätsbedingungen wird die Ausführung einer Transaktion hinsichtlich der Einhaltung von Vor- und Nachbedingungen abhängig gemacht.

# Integritätsbedingungen

## Operationale (= dynamische) Integrität

Operationale Integritätsbedingungen gewährleisten korrekte Zustandsübergänge:

- Sie überwachen die Ausführung von Operationen.
- Durch die Angabe von Integritätsbedingungen wird die Ausführung einer Transaktion hinsichtlich der Einhaltung von Vor- und Nachbedingungen abhängig gemacht.

Beispiele:

- Vom Familienstand „ledig“ ist nur ein Übergang zu verheiratet möglich – und z.B. nicht zu „geschieden“.
- Das Gehalt von Mitarbeitern darf nur steigen.

Zeitpunkt *t*:

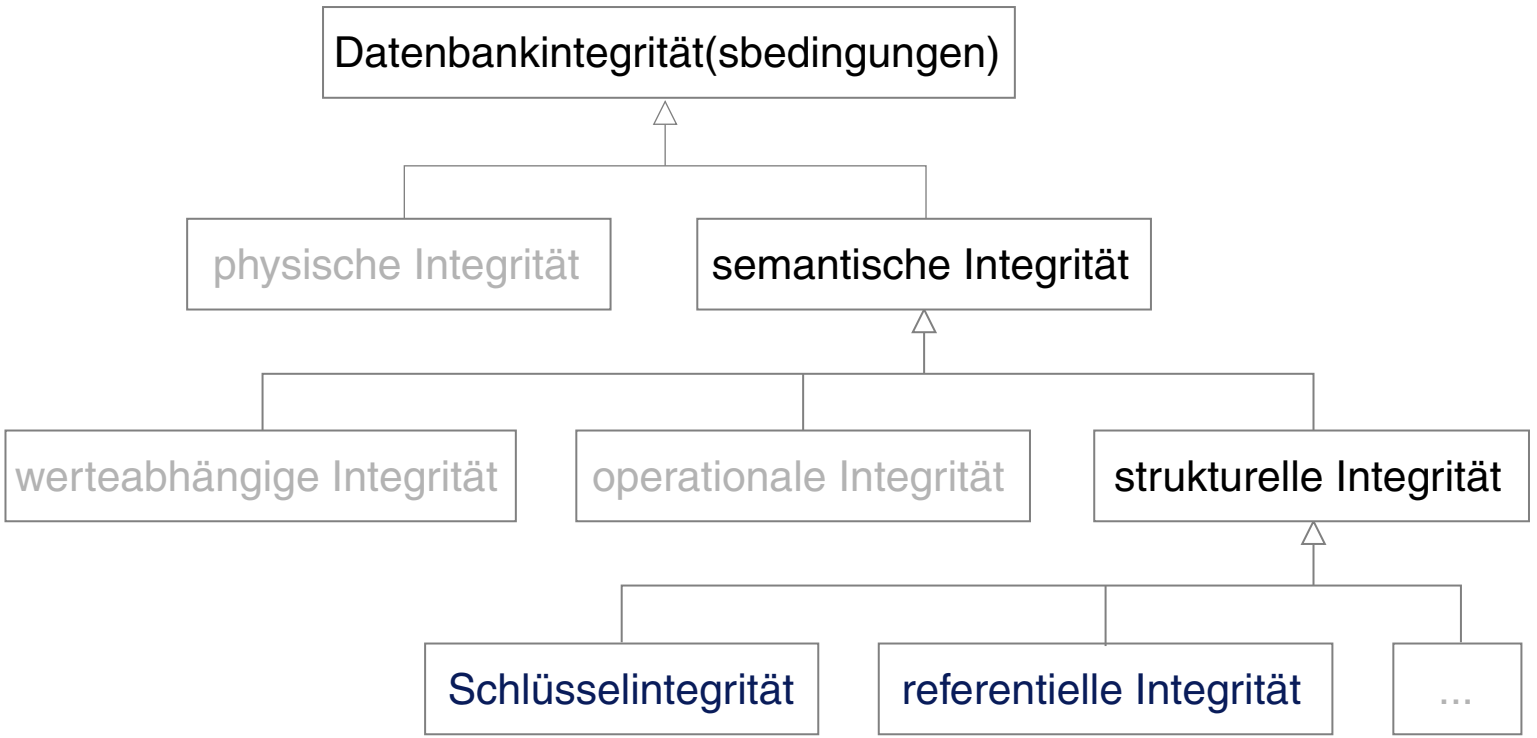
| Personal | | | |
|---|---|---|---|
| Name | PersNr. | Standort | Gehalt |
| Maier | 12 | Neuss | 66.000 |
| Steffen | 14 | Neuss | 56.000 |
| Pearl | 24 | Marl | 67.000 |
| ... | | | |

"<"

Zeitpunkt *t+1*:

| Personal | | | |
|---|---|---|---|
| Name | PersNr. | Standort | Gehalt |
| Maier | 12 | Neuss | 67.000 |
| Steffen | 14 | Neuss | 59.000 |
| Pearl | 24 | Marl | 67.000 |
| ... | | | |

# Integritätsbedingungen




[Weber 2003]

# Integritätsbedingungen

## Schlüssel und Schlüsselintegrität

**Definition 4 (Schlüssel, Primärschlüssel, Primattribute)**

Eine Attributmenge $\kappa \subseteq \mathcal{R}$ eines Relationenschemas $\mathcal{R}$ heißt identifizierende Attributmenge für eine Relation $r(\mathcal{R})$ falls gilt:

$$\forall t_1, t_2 \in r : \ t_1 \neq t_2 \quad \text{impliziert} \quad \exists A \in \kappa : t_1(A) \neq t_2(A)$$

# Integritätsbedingungen

## Schlüssel und Schlüsselintegrität

**Definition 4 (Schlüssel, Primärschlüssel, Primattribute)**

Eine Attributmenge $\kappa \subseteq \mathcal{R}$ eines Relationenschemas $\mathcal{R}$ heißt identifizierende Attributmenge für eine Relation $r(\mathcal{R})$ falls gilt:

$$\forall t_1, t_2 \in r : \; t_1 \neq t_2 \quad \text{impliziert} \quad \exists A \in \kappa : t_1(A) \neq t_2(A)$$

1. Ein Schlüssel bzw. ein Schlüsselkandidat ist eine minimale, identifizierende Attributmenge. [DB:III ER-Konzepte und ihre Semantik » Schlüssel]
2. Ein Primärschlüssel ist ein unter mehreren Schlüsselkandidaten ausgezeichneter Schlüssel.
3. Die Attribute eines Schlüssels nennt man Primattribute.

Die Schlüsselintegrität für eine Relation $r(\mathcal{R})$ fordert die Existenz eines Schlüssels für $r(\mathcal{R})$.

# Integritätsbedingungen

## Schlüssel und Schlüsselintegrität (Fortsetzung)

Beispiel:

| Personen | | | | | | | |
|---|---|---|---|---|---|---|---|
| PANr | Vorname | Nachname | PLZ | ORT | Straße | HNr. | Geb.Datum |
| 4711 | Andreas | Heuer | 18209 | DBR | BHS | 15 | 31.10.1958 |
| 5588 | Gunter | Saake | 39106 | MD | STS | 55 | 05.10.1960 |
| 6834 | Michael | Korn | 39104 | MD | BS | 41 | 24.09.1974 |
| 7754 | Andreas | Möller | 18209 | DBR | RS | 31 | 25.02.1976 |
| 8832 | Tamara | Jagellovsk | 38106 | BS | GS | 12 | 11.11.1973 |
| 9912 | Antje | Hellhof | 18059 | HRO | AES | 21 | 04.04.1970 |
| 9999 | Christa | Loeser | 69121 | HD | TS | 38 | 10.05.1969 |

- {Vorname, Nachname} und {PANr} sind Beispiele für Schlüsselkandidaten für die Relation „Personen“.
- Der Primärschlüssel wird durch Unterstreichen der entsprechenden Attribute gekennzeichnet.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität

**Definition 5 (Fremdschlüssel, referentielle Integrität)**

Seien $r_1(\mathcal{R}_1)$ und $r_2(\mathcal{R}_2)$ zwei Relationen mit den Schemata $\mathcal{R}_1$ bzw. $\mathcal{R}_2$. Sei weiterhin $\kappa$ Primärschlüssel von $\mathcal{R}_1$.

Dann ist $\alpha \subset \mathcal{R}_2$ Fremdschlüssel in $r_2$ bezüglich $\kappa$ in $r_1$, falls für alle Tupel $t_2 \in r_2$ gilt:

$$\exists t_1 \in r_1 \text{ mit } t_1(\kappa) = t_2(\alpha)$$

$$\text{bzw.} \quad \{t_2(\alpha) \mid t_2 \in r_2\} \subseteq \{t_1(\kappa) \mid t_1 \in r_1\}$$

Die Erfüllung dieser Eigenschaft heißt referentielle Integrität, Fremdschlüsselintegrität oder auch Fremdschlüsselbedingung.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität

**Definition 5 (Fremdschlüssel, referentielle Integrität)**

Seien $r_1(\mathcal{R}_1)$ und $r_2(\mathcal{R}_2)$ zwei Relationen mit den Schemata $\mathcal{R}_1$ bzw. $\mathcal{R}_2$. Sei weiterhin $\kappa$ Primärschlüssel von $\mathcal{R}_1$.

Dann ist $\alpha \subset \mathcal{R}_2$ Fremdschlüssel in $r_2$ bezüglich $\kappa$ in $r_1$, falls für alle Tupel $t_2 \in r_2$ gilt:

$$\exists t_1 \in r_1 \text{ mit } t_1(\kappa) = t_2(\alpha)$$

bzw. $$\{t_2(\alpha) \mid t_2 \in r_2\} \subseteq \{t_1(\kappa) \mid t_1 \in r_1\}$$

Die Erfüllung dieser Eigenschaft heißt referentielle Integrität, Fremdschlüsselintegrität oder auch Fremdschlüsselbedingung.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

$\mathcal{R}_2 = \{ \underbrace{B_1, B_2, B_3}, ..., B_m \}$

Fremd "schlüssel" $\alpha$

$\mathcal{R}_1 = \{ \underline{A_1}, \underline{A_2}, \underline{A_3}, ..., A_n \}$

Schlüssel $\kappa$

| $B_1$ | $B_2$ | $B_3$ | ... | $B_m$ |
|---|---|---|---|---|
| | | | | |

$r_2(\mathcal{R}_2)$

| $A_1$ | $A_2$ | $A_3$ | ... | $A_n$ |
|---|---|---|---|---|
| | | | | |

$r_1(\mathcal{R}_1)$

# Integritätsbedingungen

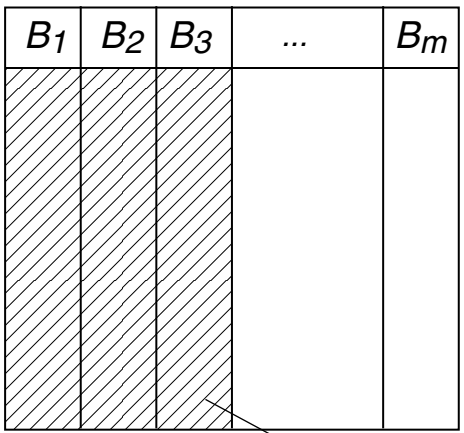
## Fremdschlüssel und referentielle Integrität (Fortsetzung)

$\mathcal{R}_2 = \{ \underbrace{B_1, B_2, B_3}, ..., B_m \}$

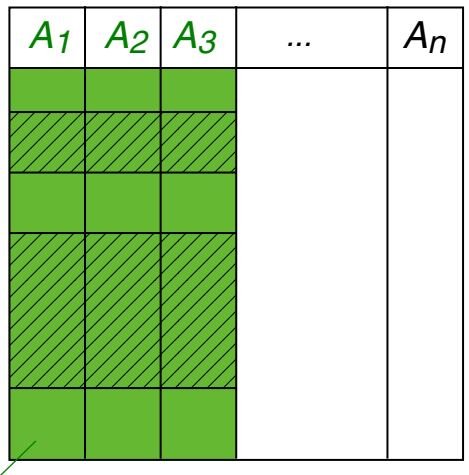
Fremd "schlüssel" $\alpha$

$\mathcal{R}_1 = \{ \underbrace{A_1, A_2, A_3}, ..., A_n \}$

Schlüssel $\kappa$




$r_2(\mathcal{R}_2)$




$r_1(\mathcal{R}_1)$

$$\{ t_2(\alpha) \mid t_2 \in r_2 \} \subseteq \{ t_1(\kappa) \mid t_1 \in r_1 \}$$

Bemerkungen:

- Jedes Teiltupel $t_2(\alpha)$ aus $r_2$ liegt als Ausprägung des Schlüssels $\kappa$ in $r_1$ vor.
- Ein Fremdschlüssel für ein Relationenschema $\mathcal{R}_2$ ist eine Attributmenge $\alpha \subset \mathcal{R}_2$, die in einem Relationenschema $\mathcal{R}_1$ Schlüssel ist. Daraus folgt $\alpha \subseteq \mathcal{R}_1$.
- Der Schlüssel $\kappa$ des Relationenschemas $\mathcal{R}_1$ wird im Relationenschema $\mathcal{R}_2$ „fremd“ verwendet. Er stellt in $\mathcal{R}_2$ in der Regel *keinen Schlüssel* dar.
- Die Attribute von Primär- und Fremdschlüssel haben jeweils dieselbe Bedeutung und, falls möglich, auch dieselbe Bezeichnung. [Heuer/Saake 2013] spricht in diesem Zusammenhang von „kompatiblen Attributlisten“.
- Notation nach [Heuer/Saake 2013] für den Sachverhalt, dass $\alpha \subset \mathcal{R}_2$ Fremdschlüssel bezüglich $\kappa$ ist: $\alpha(\mathcal{R}_2) \rightarrow \kappa(\mathcal{R}_1)$
- Eine Verschärfung der Definition der referentiellen Integrität entsteht durch die zusätzliche Forderung, dass $t_2(\alpha)$ entweder nur Nullwerte oder nur Werte ungleich Null besitzen darf. [Kemper/Eickler 2011]
- Ohne Überprüfung der referentiellen Integrität kann ein inkonsistenter Zustand der Datenbank entstehen.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Beispiel:

| Personen | | | | | | | |
|---|---|---|---|---|---|---|---|
| PANr | Vorname | Nachname | PLZ | Ort | Straße | HNr. | Geb.Datum |
| 4711 | Andreas | Heuer | 18209 | DBR | BHS | 15 | 31.10.1958 |
| 5588 | Gunter | Saake | 39106 | MD | STS | 55 | 05.10.1960 |
| 6834 | Michael | Korn | 39104 | MD | BS | 41 | 24.09.1974 |
| 7754 | Andreas | Möller | 18209 | DBR | RS | 31 | 25.02.1976 |
| 8832 | Tamara | Jagellovsk | 38106 | BS | GS | 12 | 11.11.1973 |
| 9912 | Antje | Hellhof | 18059 | HRO | AES | 21 | 04.04.1970 |
| 9999 | Christa | Loeser | 69121 | HD | TS | 38 | 10.05.1969 |

| Pers_Telefon | |
|---|---|
| PANr | Telefon |
| 4711 | 038203-12230 |
| 4711 | 0381-498-3401 |
| 4711 | 0381-498-3427 |
| 5588 | 0391-34677 |
| 5588 | 0391-5592-3800 |
| 9999 | 06221-400177 |

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Beispiel (Fortsetzung):

- {PANr} ist Schlüssel für die Relation „Personen“
  {PANr, Telefon} ist Schlüssel für die Relation „Pers_Telefon“

- {PANr} in der Relation „Pers_Telefon“ ist ein Fremdschlüssel und nimmt Bezug auf {PANr} in der Relation „Personen“.

- Fremdschlüsselbedingung:

  $$\forall t_2 \in \text{ Pers\_Telefon gilt: } \quad \exists t_1 \in \text{ Personen mit } t_1(\text{PANr}) = t_2(\text{PANr})$$

  alternativ:

  $$\{t_2(\text{PANr}) \mid t_2 \in \text{Pers\_Telefon}\} \subseteq \{t_1(\text{PANr}) \mid t_1 \in \text{Personen}\}$$

  mit den Werten des Datenbankzustands:

  $$\{4711, 5588, 9999\} \subseteq \{4711, 5588, 6834, 7754, 8832, 9912, 9999\}$$

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Sei $r_1(\mathcal{R}_1)$ eine Relation mit Primärschlüssel $\kappa$ und sei $r_2(\mathcal{R}_2)$ eine Relation mit Fremdschlüssel $\alpha$ bezüglich $\kappa$ in $r_1$. Gewährleistung der referentiellen Integrität:

1. Das Einfügen eines Tupels in $r_2$ verlangt, dass der Fremdschlüssel auf ein existierendes Tupel in $r_1$ verweist.
   Das automatische Anlegen eines Tupels in $r_1$ mit der entsprechenden Fremdschlüsselausprägung wird *Cascading Insert* genannt.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Sei $r_1(\mathcal{R}_1)$ eine Relation mit Primärschlüssel $\kappa$ und sei $r_2(\mathcal{R}_2)$ eine Relation mit Fremdschlüssel $\alpha$ bezüglich $\kappa$ in $r_1$. Gewährleistung der referentiellen Integrität:

1. Das Einfügen eines Tupels in $r_2$ verlangt, dass der Fremdschlüssel auf ein existierendes Tupel in $r_1$ verweist.
   Das automatische Anlegen eines Tupels in $r_1$ mit der entsprechenden Fremdschlüsselausprägung wird *Cascading Insert* genannt.

2. Das Ändern eines Tupels in $r_2$ verlangt, dass die neue Fremdschlüsselausprägung auf ein existierendes Tupel in $r_1$ verweist.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Sei $r_1(\mathcal{R}_1)$ eine Relation mit Primärschlüssel $\kappa$ und sei $r_2(\mathcal{R}_2)$ eine Relation mit Fremdschlüssel $\alpha$ bezüglich $\kappa$ in $r_1$. Gewährleistung der referentiellen Integrität:

1. Das Einfügen eines Tupels in $r_2$ verlangt, dass der Fremdschlüssel auf ein existierendes Tupel in $r_1$ verweist.
   Das automatische Anlegen eines Tupels in $r_1$ mit der entsprechenden Fremdschlüsselausprägung wird *Cascading Insert* genannt.

2. Das Ändern eines Tupels in $r_2$ verlangt, dass die neue Fremdschlüsselausprägung auf ein existierendes Tupel in $r_1$ verweist.

3. Das Ändern der Primärschlüsselausprägung für ein Tupel in $r_1$ verlangt, dass kein Tupel aus $r_2$ auf dieses verwiesen hat.
   Das automatische Nachvollziehen solcher Änderungen wird *Cascading Update* genannt.

# Integritätsbedingungen

## Fremdschlüssel und referentielle Integrität (Fortsetzung)

Sei $r_1(\mathcal{R}_1)$ eine Relation mit Primärschlüssel $\kappa$ und sei $r_2(\mathcal{R}_2)$ eine Relation mit Fremdschlüssel $\alpha$ bezüglich $\kappa$ in $r_1$. Gewährleistung der referentiellen Integrität:

1. Das Einfügen eines Tupels in $r_2$ verlangt, dass der Fremdschlüssel auf ein existierendes Tupel in $r_1$ verweist.
   Das automatische Anlegen eines Tupels in $r_1$ mit der entsprechenden Fremdschlüsselausprägung wird *Cascading Insert* genannt.

2. Das Ändern eines Tupels in $r_2$ verlangt, dass die neue Fremdschlüsselausprägung auf ein existierendes Tupel in $r_1$ verweist.

3. Das Ändern der Primärschlüsselausprägung für ein Tupel in $r_1$ verlangt, dass kein Tupel aus $r_2$ auf dieses verwiesen hat.
   Das automatische Nachvollziehen solcher Änderungen wird *Cascading Update* genannt.

4. Das Löschen eines Tupels in $r_1$ verlangt, dass kein Tupel aus $r_2$ auf dieses verwiesen hat.
   Das Verhindern der Löschung von Tupeln aus $r_1$ solange noch hierauf bezugnehmende Tupel in $r_2$ vorhanden sind, wird *Prohibited Delete* genannt. Die Propagierung von Löschungen in $r_1$ weiter zu $r_2$ wird als *Cascading Delete* bezeichnet.

# Integritätsbedingungen

**Definition 6 (Lokale und globale Integritätsbedingung)**

Eine lokale Integritätsbedingung $b$ ist eine Bool'sche Funktion, die auf der Menge der Relationen eines Relationenschemas $\mathcal{R}$ definiert ist:

$$b : \{r \mid r(\mathcal{R})\} \rightarrow \{\text{true}, \text{false}\}$$

Eine globale Integritätsbedingung $b$ ist eine Bool'sche Funktion, die auf der Menge der Datenbanken eines Datenbankschemas $\boldsymbol{\mathcal{R}}$ definiert ist:

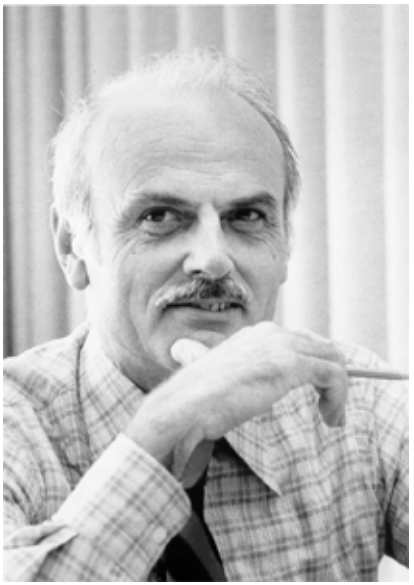
$$b : \{d \mid d(\boldsymbol{\mathcal{R}})\} \rightarrow \{\text{true}, \text{false}\}$$

Bemerkungen:

- Ein Schlüssel bzw. die Schlüsselintegrität ist eine lokale Integritätsbedingung.
- Ein Fremdschlüssel bzw. die referentielle Integrität ist eine globale Integritätsbedingung.

# Das relationale Modell

## Historie



[E. F. Codd, *1923, † 2003]

Codd, E.: *A Relational Model of Data for Large Shared Databanks.*
Communications of the ACM, Vol. 13, No. 6, 377-387, 1970.

- Das relationale Modell ist von E. F. Codd im Jahre 1970 vorgestellt worden.
- Mittlerweile ist es das am weitesten verbreitete Datenbankmodell.
- 1981 erhielt E. F. Codd den Turing Award für seinen fundamentalen und fortwährenden Beitrag zur Theorie und Praxis von Datenbanksystemen.

Bemerkungen:

- Dr. E.F. Codd, an IBM researcher, first developed the relational data model in 1970.
- In 1985, Dr. Codd published a list of 12 rules that concisely define an ideal relational database, which have provided a guideline for the design of all relational database systems ever since. [www.itworld.com 2002]

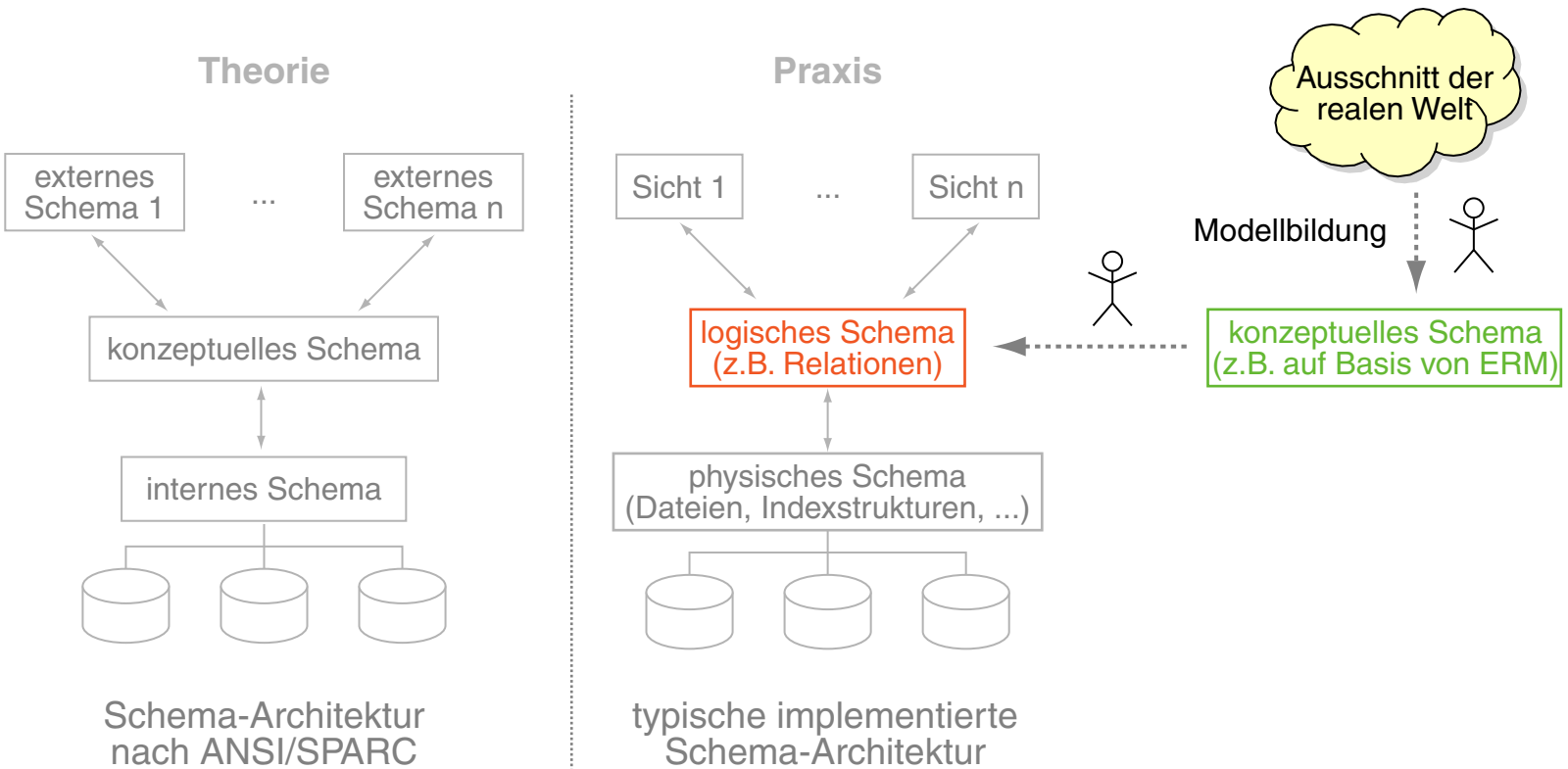
# Kapitel DB:IV (Fortsetzung)

## IV. Logischer Datenbankentwurf mit dem relationalen Modell

- Das relationale Modell
- Integritätsbedingungen
- Umsetzung ER-Schema in relationales Schema

# Umsetzung ER-Schema in relationales Schema

## Einordnung



# Umsetzung ER-Schema in relationales Schema

## Einordnung (Fortsetzung)

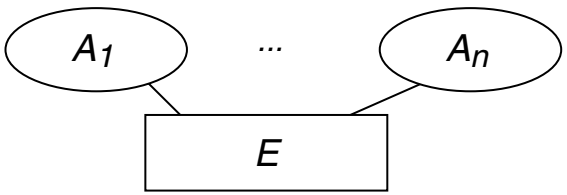
Das ER-Modell besitzt zwei grundlegende Strukturierungskonzepte:

1. Entity-Typen $E(A_1, \ldots, A_n)$
2. Beziehungstypen $R(E_1, \ldots, E_m;\ A_1, \ldots, A_n)$

Im relationalen Modell werden beide auf das einzige Strukturierungskonzept „Relationenschema“, $\mathcal{R}$, abgebildet. Hierbei dient das Konzept der Fremdschlüssel zur Abbildung von Beziehungstypen.

# Umsetzung ER-Schema in relationales Schema

## Reguläre Entity-Typen

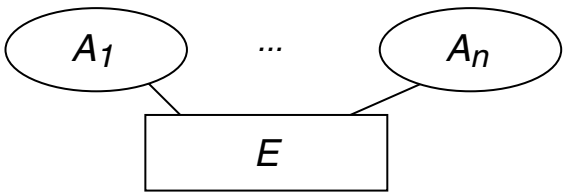



$$\mathcal{R}_E = \{\underline{ID}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa$ bzw. $\{ID\}$

# Umsetzung ER-Schema in relationales Schema

## Reguläre Entity-Typen




$$\mathcal{R}_E = \{\underline{ID}, A_1, \ldots, A_n\}$$

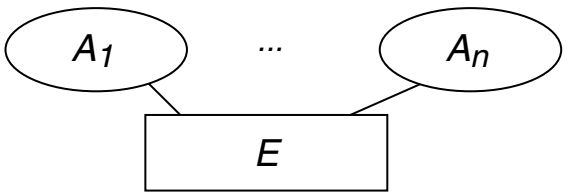
Schlüssel: $\kappa$ bzw. $\{ID\}$

Umsetzung:

1. Dem Entity-Typ $E$ wird Relationenschema $\mathcal{R}_E$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $E$ werden Attribute von $\mathcal{R}_E$.

# Umsetzung ER-Schema in relationales Schema

## Reguläre Entity-Typen




$$\mathcal{R}_E = \{\underline{ID}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa$ bzw. $\{ID\}$

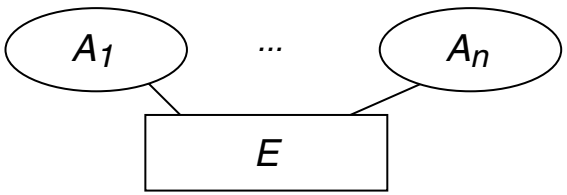
Umsetzung:

1. Dem Entity-Typ $E$ wird Relationenschema $\mathcal{R}_E$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $E$ werden Attribute von $\mathcal{R}_E$.
2. Der Schlüssel $\kappa \subseteq \{A_1, \ldots, A_n\}$ von $E$ wird Primärschlüssel von $\mathcal{R}_E$.

   Alternative:
   Festlegen eines formalen Primärschlüssels durch Hinzufügen eines Schlüsselattributes *ID* zur Umsetzung der Eindeutigkeit von Entitäten. Der ursprüngliche Schlüssel $\kappa$ ist dann ein weiterer Schlüssel im Relationenschema $\mathcal{R}_E$.

# Umsetzung ER-Schema in relationales Schema

## Reguläre Entity-Typen




$$\mathcal{R}_E = \{\underline{ID}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa$ bzw. $\{ID\}$

Umsetzung:

1. Dem Entity-Typ $E$ wird Relationenschema $\mathcal{R}_E$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $E$ werden Attribute von $\mathcal{R}_E$.
2. Der Schlüssel $\kappa \subseteq \{A_1, \ldots, A_n\}$ von $E$ wird Primärschlüssel von $\mathcal{R}_E$.

   Alternative:
   Festlegen eines formalen Primärschlüssels durch Hinzufügen eines Schlüsselattributes *ID* zur Umsetzung der Eindeutigkeit von Entitäten. Der ursprüngliche Schlüssel $\kappa$ ist dann ein weiterer Schlüssel im Relationenschema $\mathcal{R}_E$.
3. Der Primärschlüssel wird durch Unterstreichen gekennzeichnet.

Bemerkungen:

- Die Bezeichnung „regulärer Entity-Typ“ dient als Unterscheidung zu
  - abhängigen bzw. schwachen Entity-Typen sowie zu den
  - spezialisierten Entity-Typen, die in einer IST-Beziehung stehen.

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen

Zwei Umsetzungsstrategien:

(a) Direkte Abbildung auf ein adäquates Schema.

(b) Kanonische Umsetzung („Cross-Reference“) mit anschließender Zusammenfassung von Relationenschemata.

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen

Zwei Umsetzungsstrategien:

(a) Direkte Abbildung auf ein adäquates Schema.

(b) Kanonische Umsetzung („Cross-Reference“) mit anschließender Zusammenfassung von Relationenschemata.

Besondere Behandlung für folgende Fälle:

1. 1:n-Beziehung (Formalismus I für Kardinalitäten)
2. 1:1-Beziehung (Formalismus I für Kardinalitäten)
3. [0,1] und [1,1] bei $[min, max]$-Beschränkung (Formalismus II für Kardinalitäten)
4. existenzabhängige (schwache) Entity-Typen
5. IST-Beziehungstypen
6. rekursive Beziehungstypen

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen: Kapazitätserhaltung

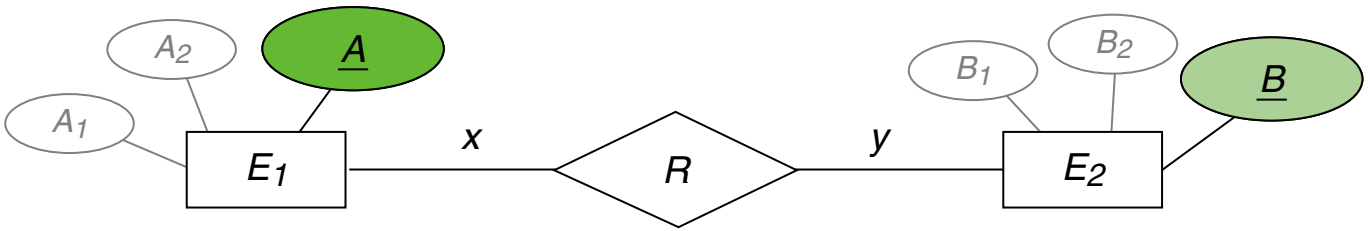
Eine zentrale Forderung bei der Abbildung von Beziehungstypen ist die Kapazitätserhaltung: alle Instanzen des ER-Modells sind auch Instanzen des relationalen Modells und umgekehrt.

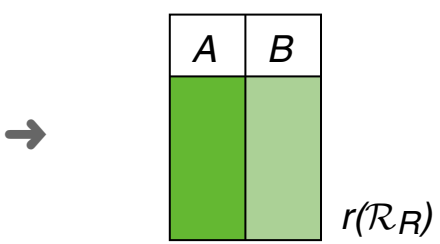
**Definition 7 (kapazitätserhaltend)**

Gibt es eine bijektive Abbildung zwischen den Instanzen eines Entity-Relationship-Modells und den Instanzen eines relationalen Modells, so nennt man die Transformation zwischen den Modellen kapazitätserhaltend.

# Umsetzung ER-Schema in relationales Schema

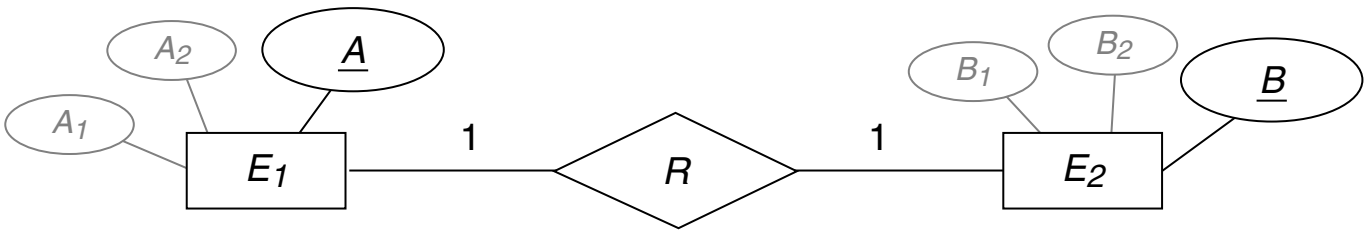
## Beziehungstypen: Kapazitätserhaltung (Fortsetzung)



# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen: Kapazitätserhaltung (Fortsetzung)



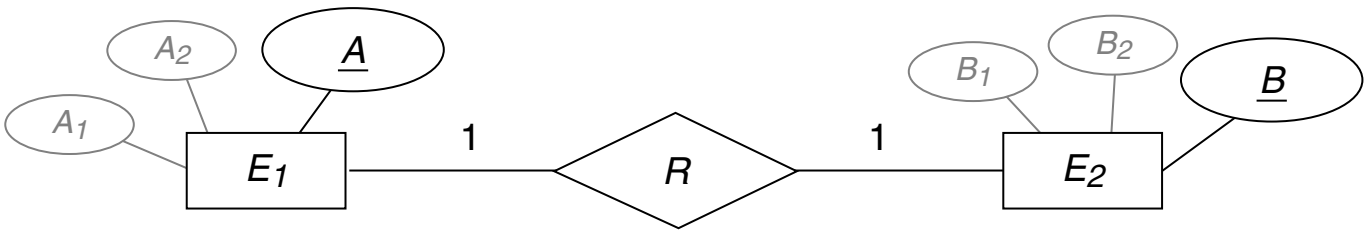

1:1-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen: Kapazitätserhaltung (Fortsetzung)




1:1-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

mögliche Relationen:

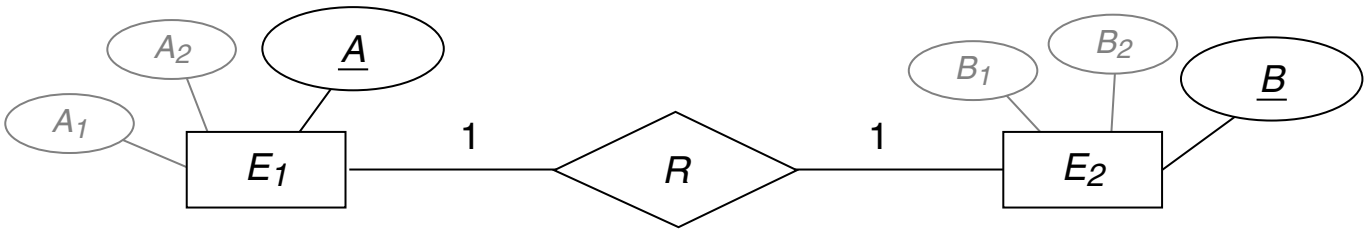
$r_1(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_2)\}$

$r_2(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

(kapazitätserhöhend)

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen: Kapazitätserhaltung (Fortsetzung)




1:1-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

Modellierung (b)

$\mathcal{R}_R = \{A, B\}$ mit *zwei* Schlüsseln $\{\underline{A}\}, \{B\}$

mögliche Relationen:

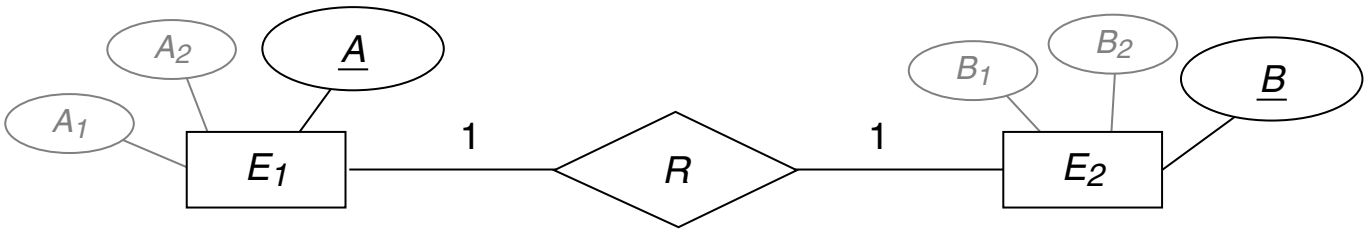
$r_1(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_2)\}$

$r_2(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

(kapazitätserhöhend)

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen: Kapazitätserhaltung (Fortsetzung)




1:1-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

mögliche Relationen:

$r_1(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_2)\}$

$r_2(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

(kapazitätserhöhend)

Modellierung (b)

$\mathcal{R}_R = \{A, B\}$ mit *zwei* Schlüsseln $\{\underline{A}\}, \{B\}$

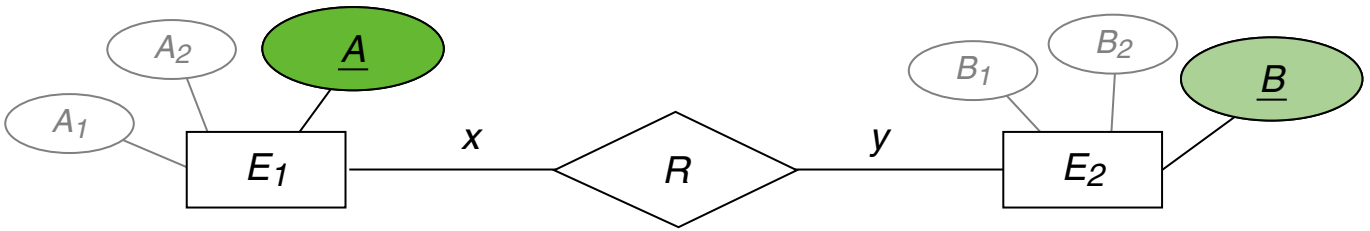
mögliche Relation:

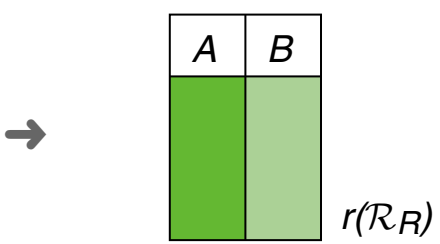
$r(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_2)\}$

(kapazitätserhaltend)

# Umsetzung ER-Schema in relationales Schema

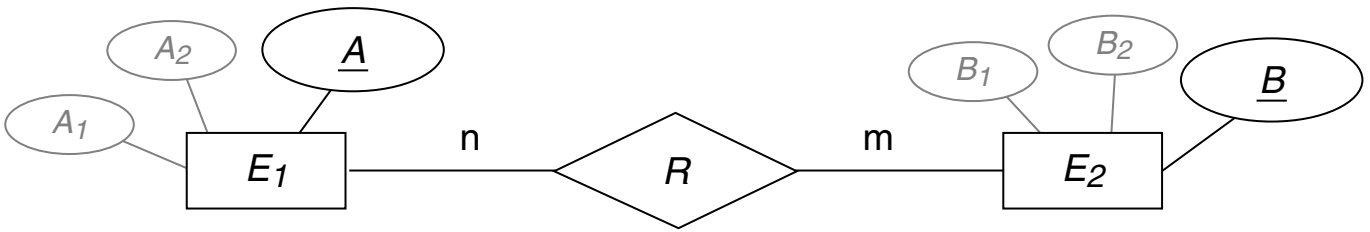
## Kapazitätserhaltung (Fortsetzung)



# Umsetzung ER-Schema in relationales Schema

## Kapazitätserhaltung (Fortsetzung)



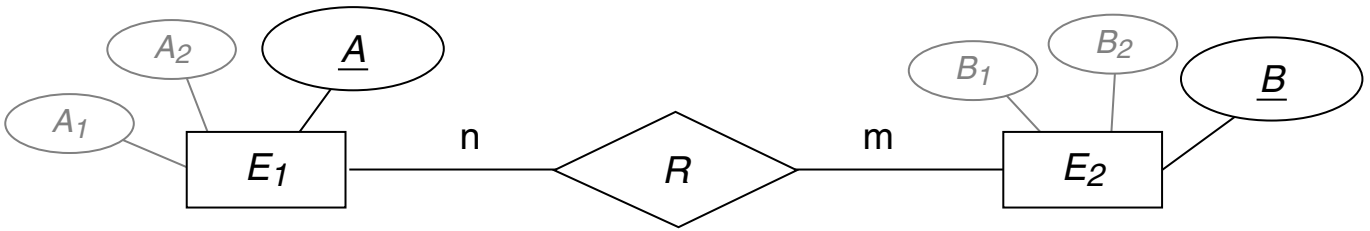

n:m-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

# Umsetzung ER-Schema in relationales Schema

## Kapazitätserhaltung (Fortsetzung)




n:m-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

mögliche Relation:

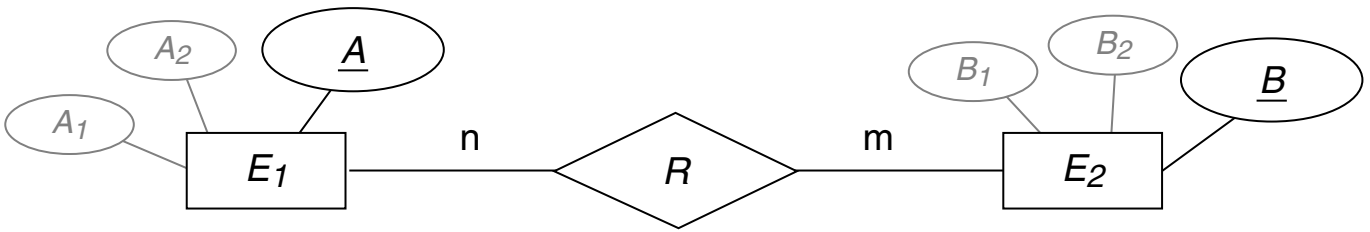
$r(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

~~$r(\mathcal{R}_R) = \{(a_1, b_1), (a_1, b_2), (a_2, b_2)\}$~~

(kapazitätsvermindernd)

# Umsetzung ER-Schema in relationales Schema

## Kapazitätserhaltung (Fortsetzung)




n:m-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

Modellierung (b)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A, B\}$

mögliche Relation:

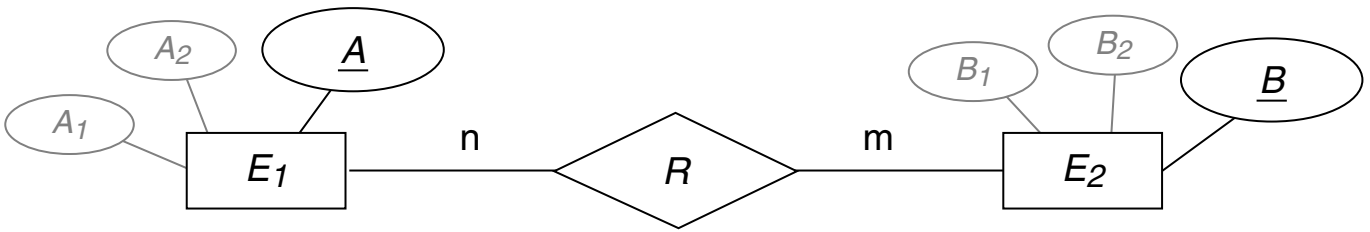
$r(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

~~$r(\mathcal{R}_R) = \{(a_1, b_1), (a_1, b_2), (a_2, b_2)\}$~~

(kapazitätsvermindernd)

# Umsetzung ER-Schema in relationales Schema

## Kapazitätserhaltung (Fortsetzung)




n:m-Beziehung

Modellierung (a)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A\}$

mögliche Relation:

$r(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_1)\}$

~~$r(\mathcal{R}_R) = \{(a_1, b_1), (a_1, b_2), (a_2, b_2)\}$~~

(kapazitätsvermindernd)

Modellierung (b)

$\mathcal{R}_R = \{A, B\}$ mit Schlüssel $\{A, B\}$

mögliche Relationen:

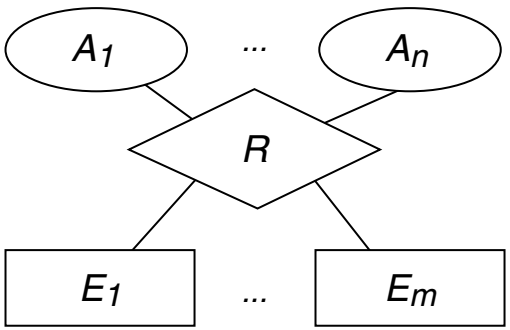
$r_1(\mathcal{R}_R) = \{(a_1, b_1), (a_2, b_2)\}$

$r_2(\mathcal{R}_R) = \{(a_1, b_1), (a_1, b_2), (a_2, b_2)\}$

(kapazitätserhaltend)

# Umsetzung ER-Schema in relationales Schema

## Reguläre Beziehungstypen

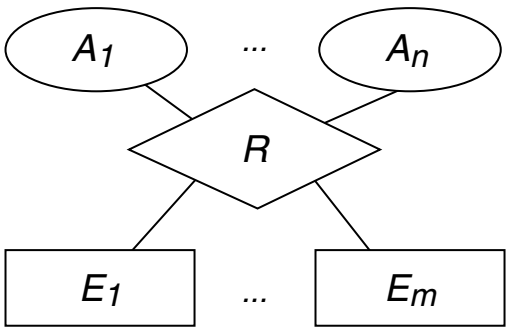



$$\mathcal{R}_R = \{ID_1, \ldots, ID_m, A_1, \ldots, A_n\}$$

Schlüssel: $\alpha \subseteq \kappa_1 \cup \ldots \cup \kappa_m$ bzw.
$\alpha \subseteq \{ID_1, \ldots, ID_m\}$

# Umsetzung ER-Schema in relationales Schema

## Reguläre Beziehungstypen




$$\mathcal{R}_R = \{ID_1, \ldots, ID_m, A_1, \ldots, A_n\}$$

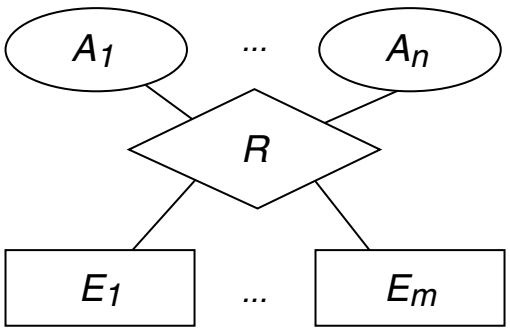
Schlüssel: $\alpha \subseteq \kappa_1 \cup \ldots \cup \kappa_m$ bzw.
$\alpha \subseteq \{ID_1, \ldots, ID_m\}$

Cross-Reference [Elmasri/Navathe 2010]:

1. Dem Beziehungstyp $R$ wird Relationenschema $\mathcal{R}_R$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $R$ werden Attribute von $\mathcal{R}_R$.

# Umsetzung ER-Schema in relationales Schema

## Reguläre Beziehungstypen




$$\mathcal{R}_R = \{ID_1, \ldots, ID_m, A_1, \ldots, A_n\}$$

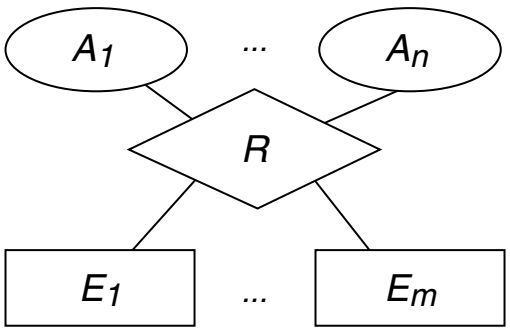
Schlüssel: $\alpha \subseteq \kappa_1 \cup \ldots \cup \kappa_m$ bzw.
$\alpha \subseteq \{ID_1, \ldots, ID_m\}$

Cross-Reference [Elmasri/Navathe 2010]:

1. Dem Beziehungstyp $R$ wird Relationenschema $\mathcal{R}_R$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $R$ werden Attribute von $\mathcal{R}_R$.
2. Die Attribute in den $\kappa_i$ (bzw. die $ID_i$) von $\mathcal{R}_{E_i}$ werden Attribute von $\mathcal{R}_R$.

# Umsetzung ER-Schema in relationales Schema

## Reguläre Beziehungstypen




$$\mathcal{R}_R = \{ID_1, \ldots, ID_m, A_1, \ldots, A_n\}$$

Schlüssel: $\alpha \subseteq \kappa_1 \cup \ldots \cup \kappa_m$ bzw.
$\alpha \subseteq \{ID_1, \ldots, ID_m\}$

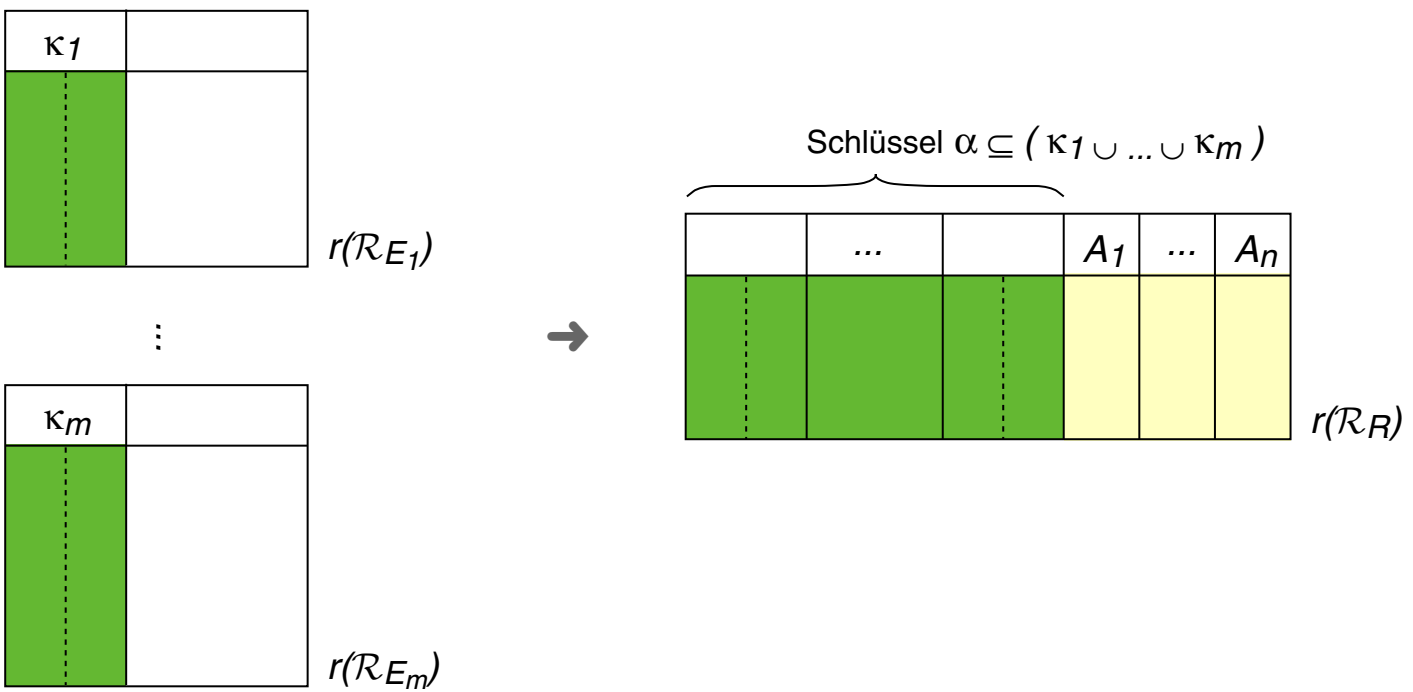
Cross-Reference [Elmasri/Navathe 2010]:

1. Dem Beziehungstyp $R$ wird Relationenschema $\mathcal{R}_R$ zugeordnet.
   Die Attribute $A_1, \ldots, A_n$ von $R$ werden Attribute von $\mathcal{R}_R$.
2. Die Attribute in den $\kappa_i$ (bzw. die $ID_i$) von $\mathcal{R}_{E_i}$ werden Attribute von $\mathcal{R}_R$.
3. Der Schlüssel von $\mathcal{R}_R$ ist eine Teilmenge der Vereinigungsmenge der $\kappa_i$ (bzw. der Menge aller $ID_i$).

# Umsetzung ER-Schema in relationales Schema

## Reguläre Beziehungstypen (Fortsetzung)

Cross-Reference:



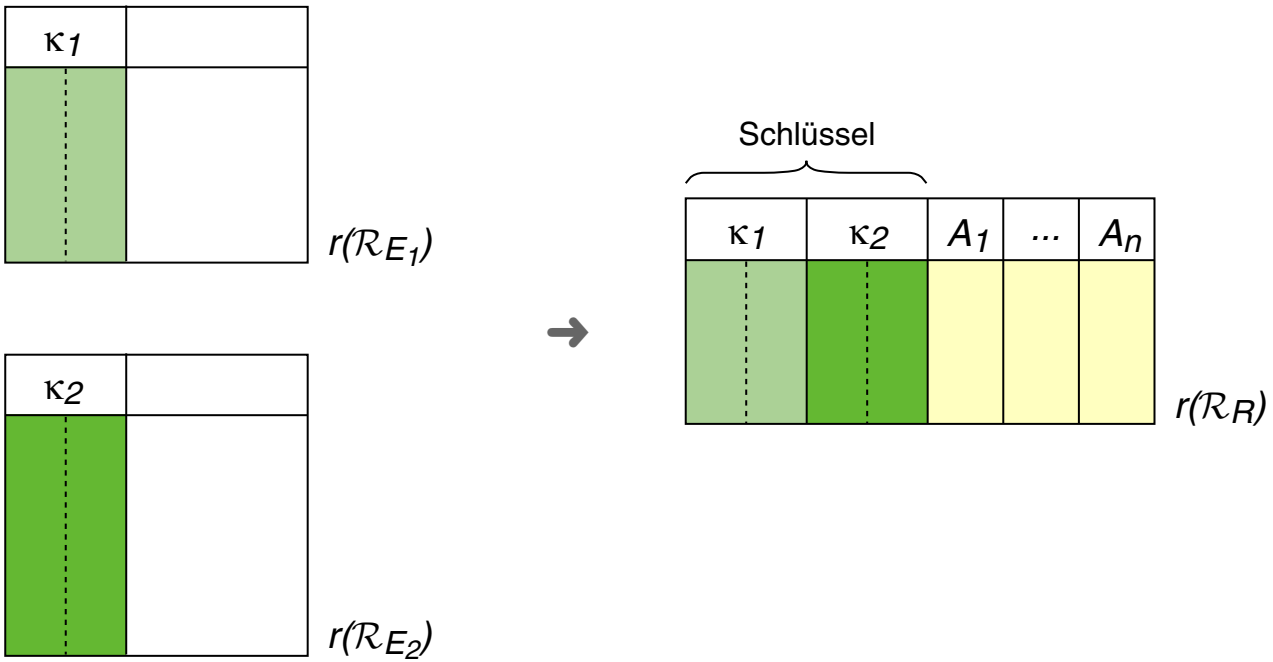
Bemerkungen:

- Die Bezeichnung „regulärer Beziehungstyp“ dient als Unterscheidung zu
  - Beziehungstypen zu abhängigen bzw. schwachen Entity-Typen sowie zu
  - IST-Beziehungstypen.
- Die $\kappa_i \subset \mathcal{R}_R$ (bzw. die $\{ID_i\} \subset \mathcal{R}_R$) sind Fremdschlüssel in $\mathcal{R}_R$ bzgl. $\kappa_i$ (bzw. $\{ID_i\}$) in $\mathcal{R}_{E_i}$.
- Es stellt sich die Frage, wie die Teilmenge aus der Vereinigungsmenge der $\kappa_i$ gebildet wird, so dass ein Schlüssel für die Relation $\mathcal{R}_R$ entsteht. Man kann diese Frage nicht in der Allgemeinheit beantworten.
  Vergleiche hierzu die möglichen funktionalen Beziehungen, die beispielsweise von einer $x : y : z$ -Relation, $x, y, z \in \{1, \mathrm{n}, \mathrm{m}\}$, impliziert sein können: falls keine funktionale Beziehung gegeben ist, also $x$ und $y$ und $z \neq 1$, so bilden nur alle Schlüsselattribute der drei Entity-Typen zusammen einen Schlüssel für $\mathcal{R}_R$. Gibt es einen funktionalen Zusammenhang, also $x$ oder $y$ oder $z = 1$, so bildet die Vereinigungsmenge der Schlüsselattribute der beiden Entity-Typen des Urbildbereiches einen Schlüssel für $\mathcal{R}_R$.

# Umsetzung ER-Schema in relationales Schema

## n:m-Beziehungstypen

Cross-Reference:



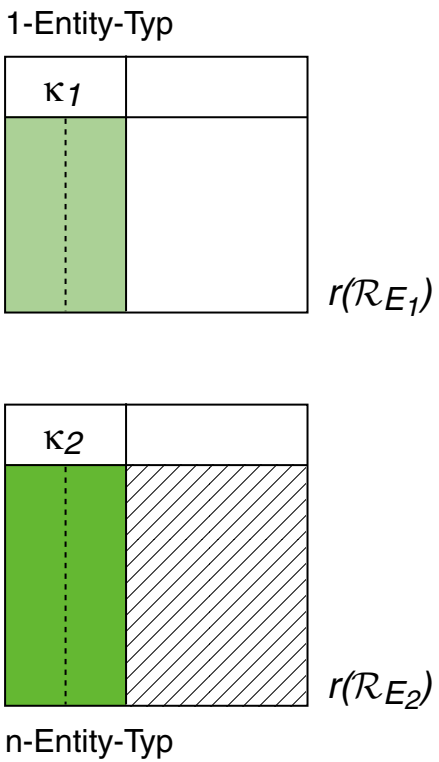

Die Primärschlüssel der beteiligten Relationenschemata $\mathcal{R}_{E_1}$ und $\mathcal{R}_{E_2}$ bilden zusammen den Schlüssel im Relationenschema $\mathcal{R}_R$ des n:m-Beziehungstyps.

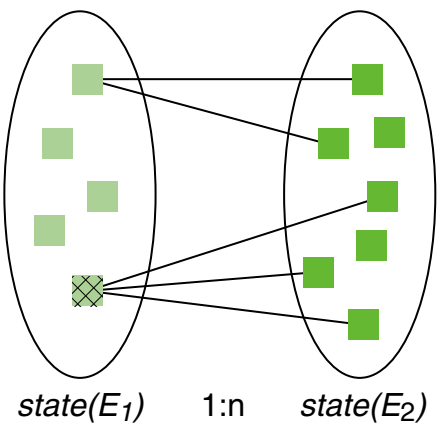
[Kapazitätserhaltung]

# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen

Funktionaler Zusammenhang $E_2 \rightarrow E_1$:

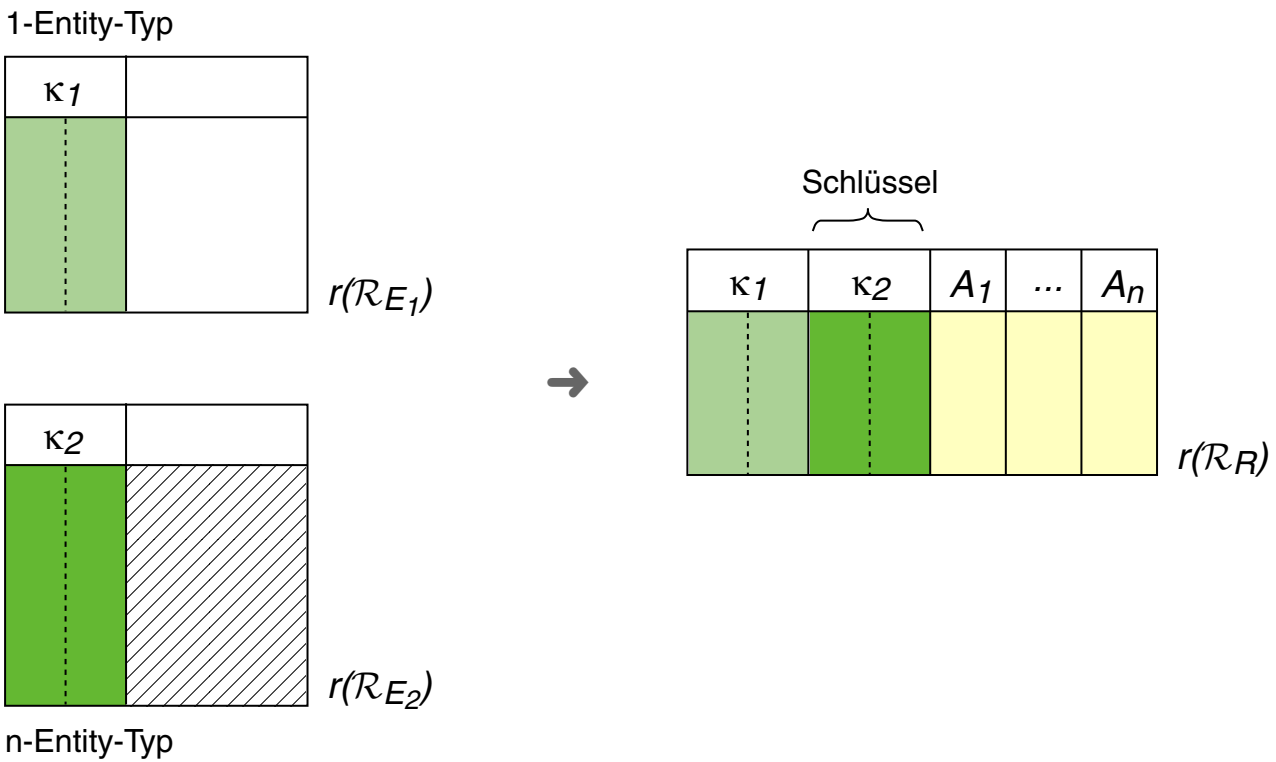


# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen (Fortsetzung)

Cross-Reference:



# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen (Fortsetzung) [Sonderfall 1]

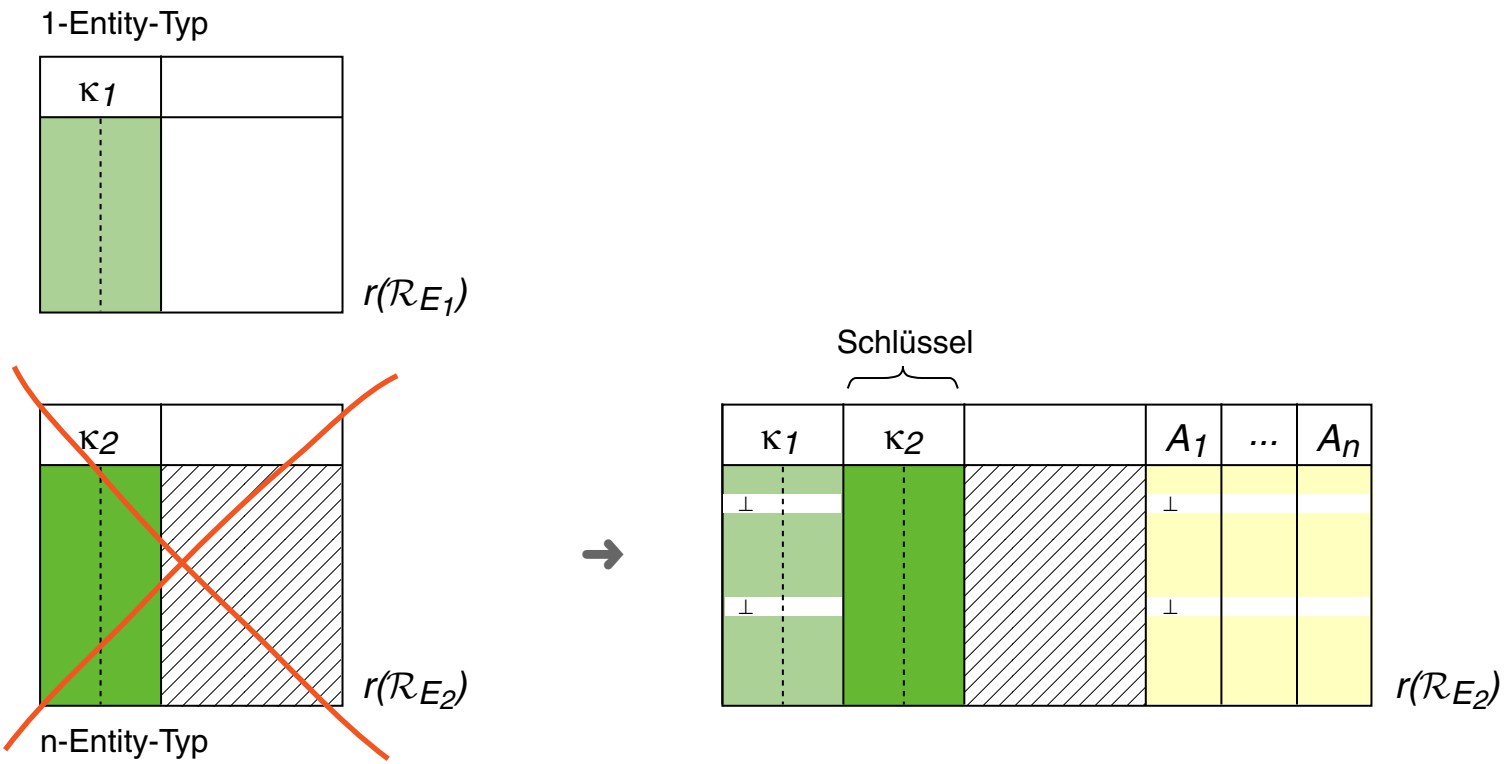
Als Verfeinerung der Cross-Reference kann man bei 1:n-Beziehungstypen das Relationenschema $\mathcal{R}_R$ mit dem Relationenschema $\mathcal{R}_{E_2}$, das den n-Entity-Typ im 1:n-Beziehungstyp repräsentiert, zusammenfassen:

1. Die Attribute des Primärschlüssels in $\mathcal{R}_{E_1}$ werden Attribute in $\mathcal{R}_{E_2}$ und stellen dort einen entsprechenden Fremdschlüssel dar.
2. Die Attribute des 1:n-Beziehungstyps werden Attribute in $\mathcal{R}_{E_2}$.
3. Der Primärschlüssel des n-Entity-Typs wird Schlüssel im zusammengefassten Relationenschema.

# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen (Fortsetzung)

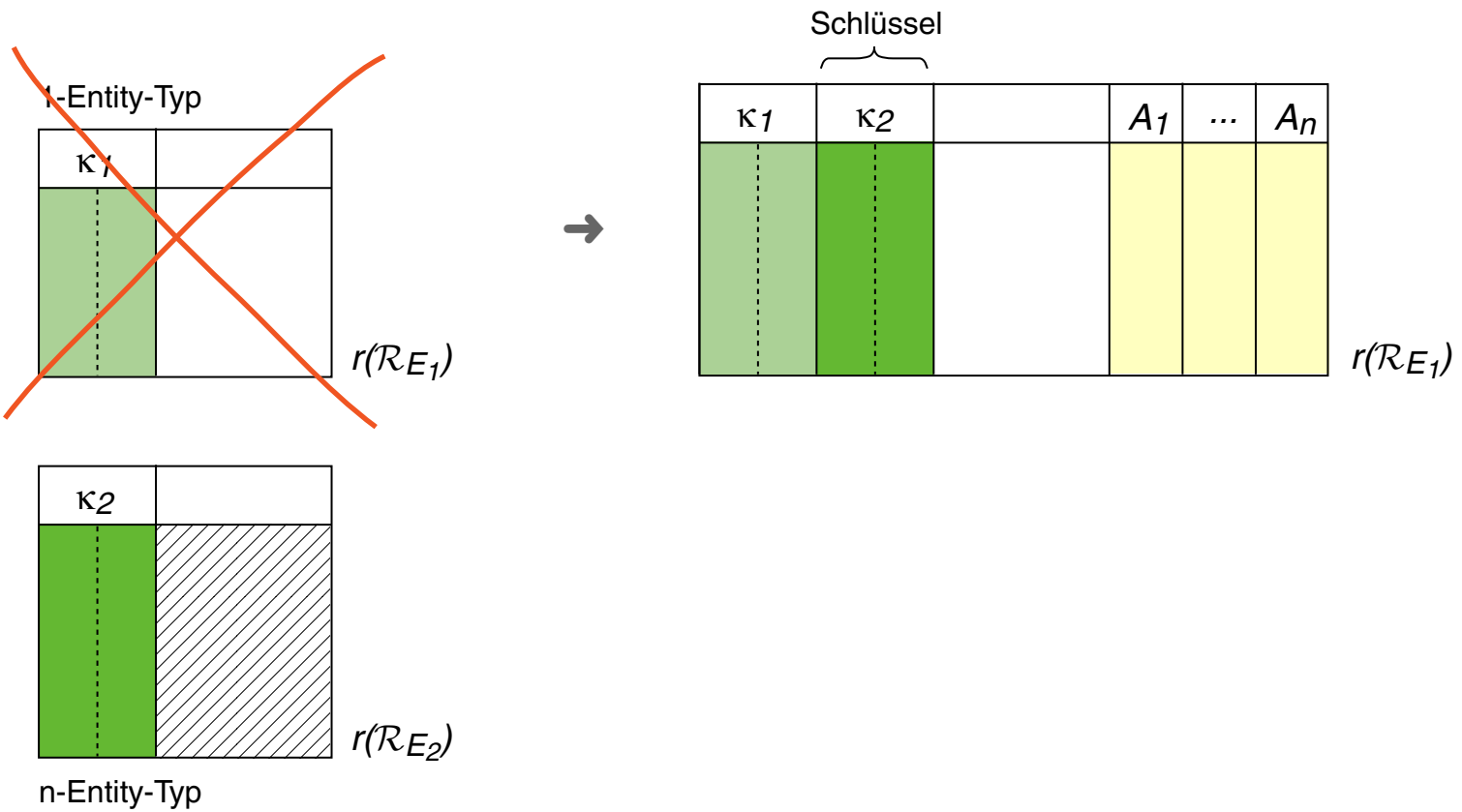
Erlaubte Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_2}$:



# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen (Fortsetzung)

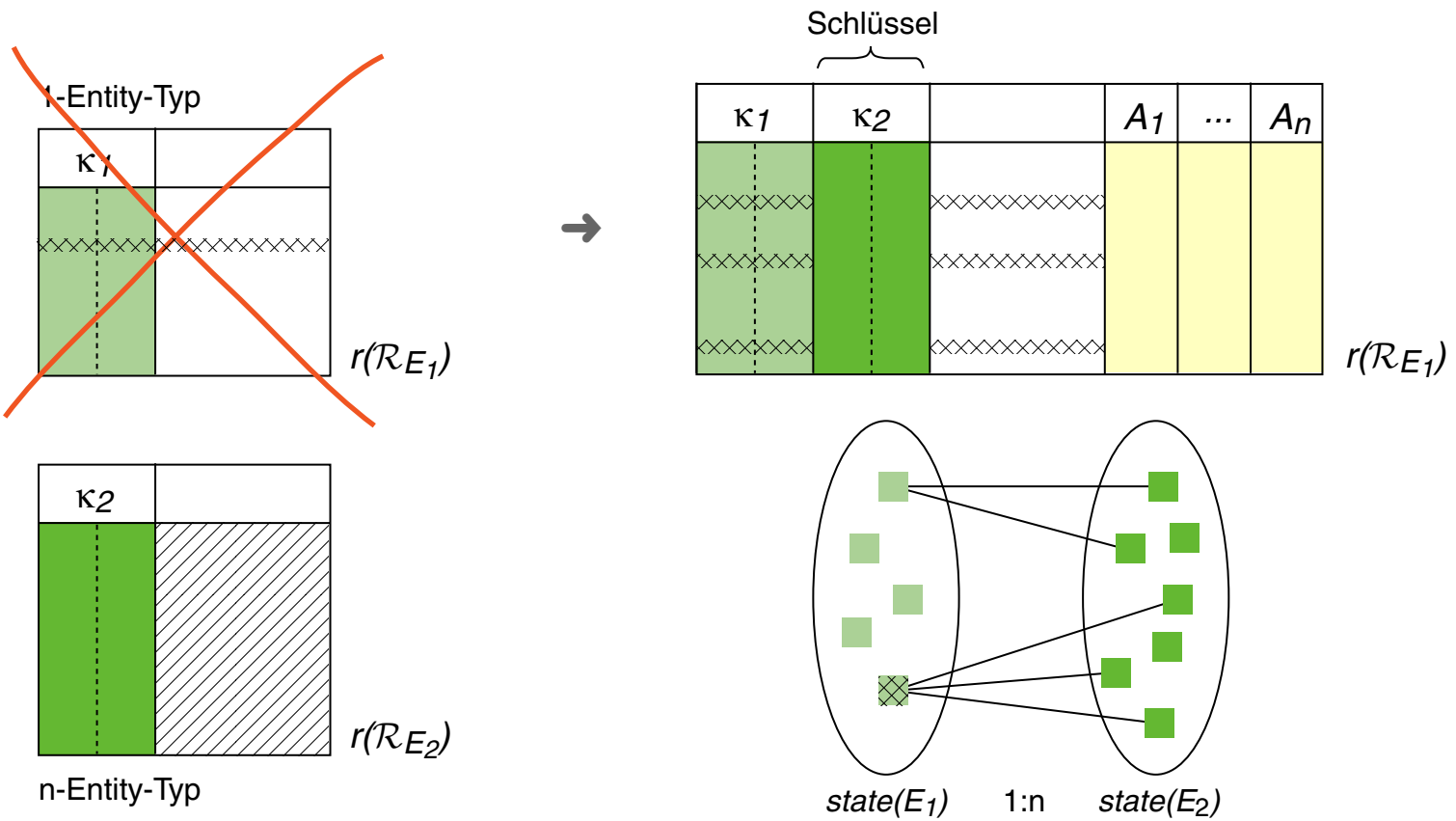
Unerlaubte Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_1}$:



# Umsetzung ER-Schema in relationales Schema

## 1:n-Beziehungstypen (Fortsetzung)

Unerlaubte Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_1}$:



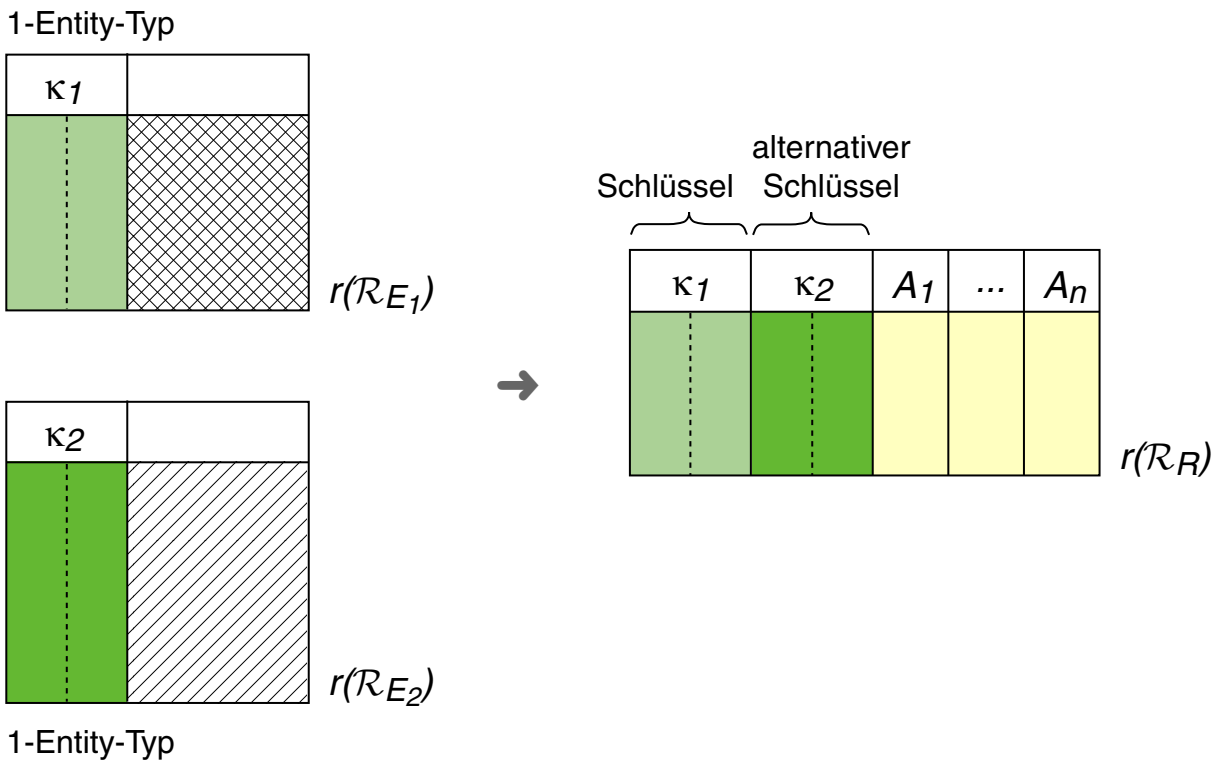
Bemerkungen:

- [Kemper/Eickler 2011] gibt folgende Regel als Hilfe bei der Zusammenfassung von Relationen an: „Nur Relationen mit gleichem Schlüssel zusammenfassen.“
  In der Illustration sind das die beiden Relationen $\mathcal{R}_R$ und $\mathcal{R}_{E_2}$; beide haben den Schlüssel $\kappa_2$.

# Umsetzung ER-Schema in relationales Schema

## 1:1-Beziehungstypen [Kapazitätserhaltung]

Cross-Reference:



# Umsetzung ER-Schema in relationales Schema

## 1:1-Beziehungstypen (Fortsetzung) [Sonderfall 2]

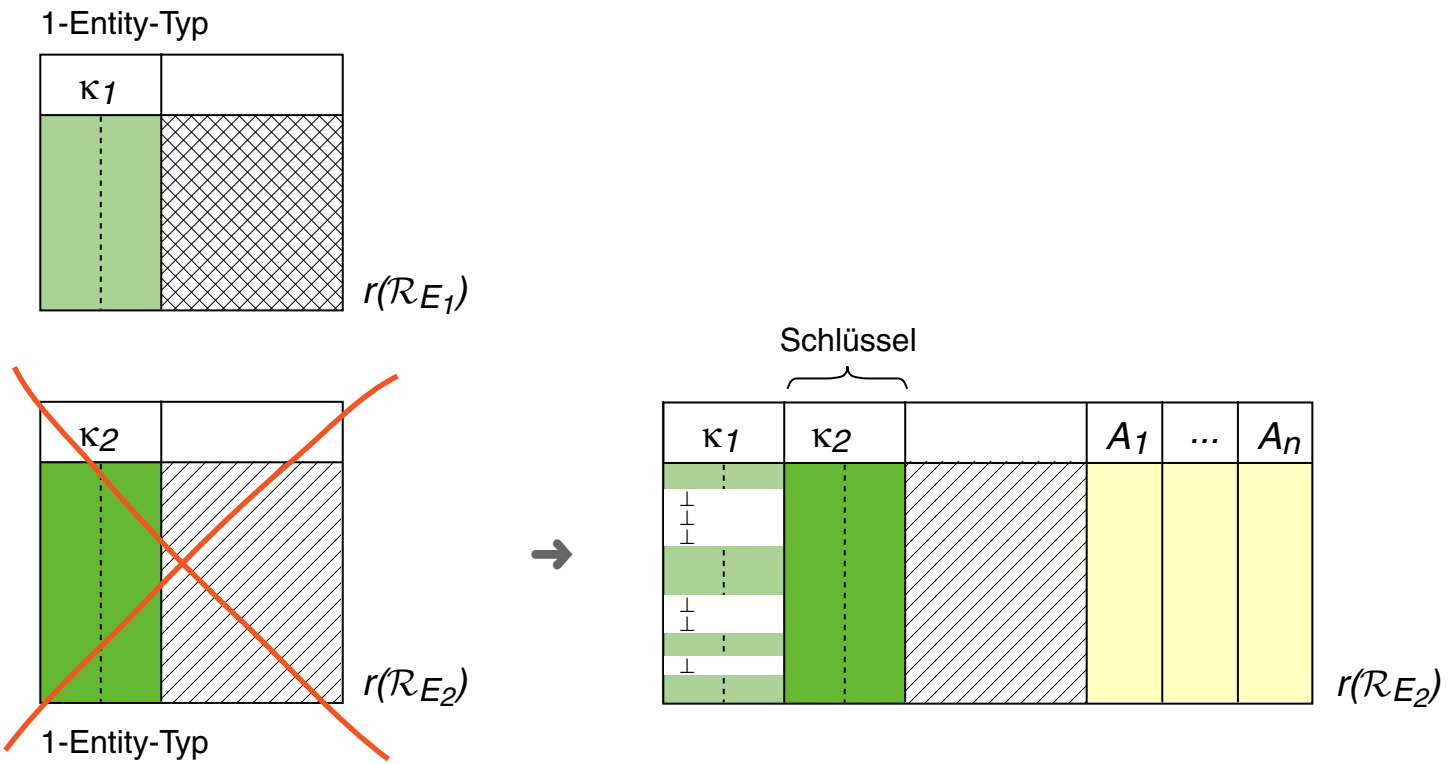
Als Verfeinerung der Cross-Reference kann man bei 1:1-Beziehungstypen das Relationenschema $\mathcal{R}_R$ mit einem der beiden Relationenschemata der beteiligten Entity-Typen, $\mathcal{R}_{E_2}$ oder $\mathcal{R}_{E_1}$, zusammenfassen:

1. Die Attribute des Primärschlüssels in $\mathcal{R}_{E_1}$ $(\mathcal{R}_{E_2})$ werden Attribute in $\mathcal{R}_{E_2}$ $(\mathcal{R}_{E_1})$ und stellen dort einen entsprechenden Fremdschlüssel dar.
2. Die Attribute des 1:1-Beziehungstyps werden Attribute in $\mathcal{R}_{E_2}$ $(\mathcal{R}_{E_1})$.
3. Der Primärschlüssel von $\mathcal{R}_{E_2}$ $(\mathcal{R}_{E_1})$ wird Schlüssel im zusammengefassten Relationenschema.

# Umsetzung ER-Schema in relationales Schema

## 1:1-Beziehungstypen (Fortsetzung)

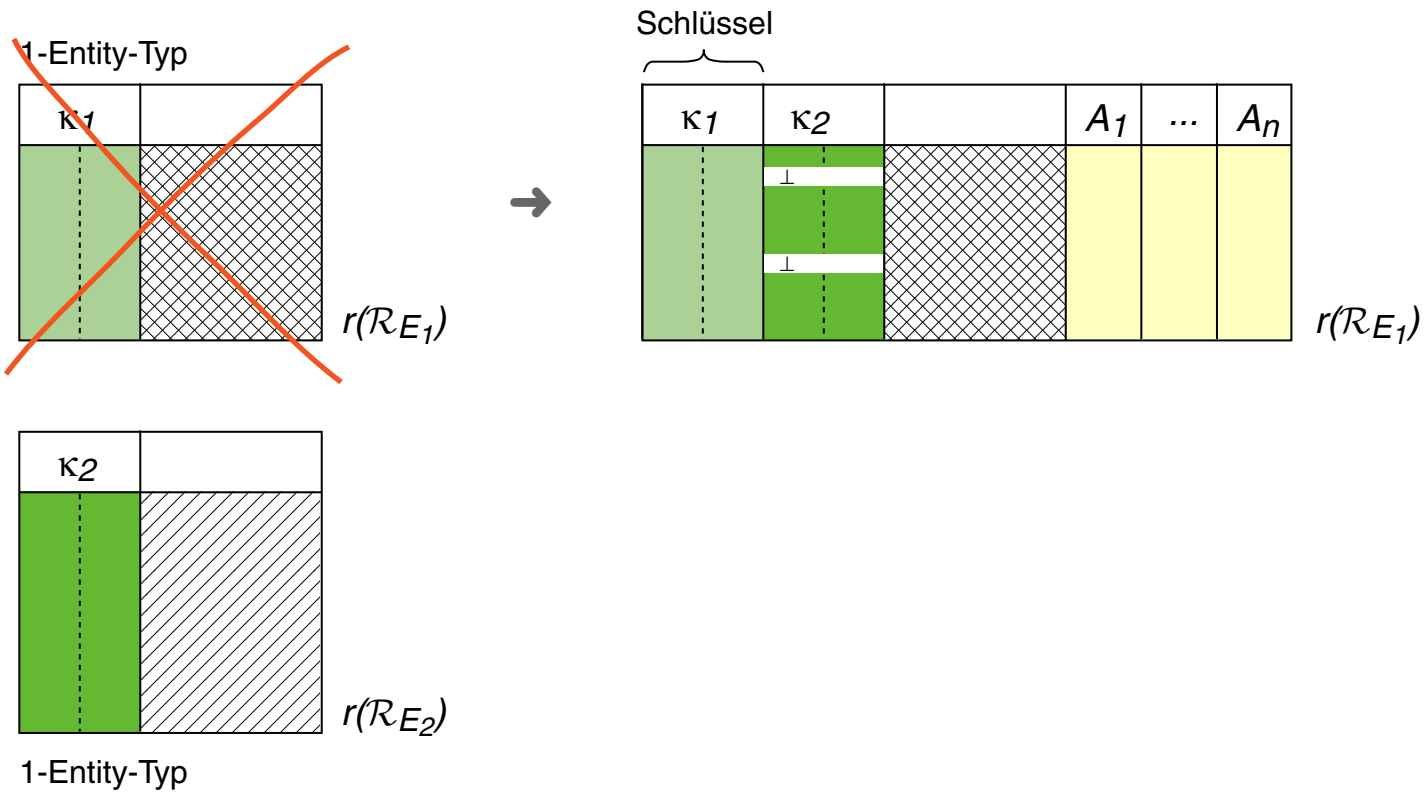
Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_2}$:



# Umsetzung ER-Schema in relationales Schema

## 1:1-Beziehungstypen (Fortsetzung)

Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_1}$:



Bemerkungen:

- Ob man den Primärschlüssel von $\mathcal{R}_{E_1}$ in $\mathcal{R}_{E_2}$ integriert – oder umgekehrt, sollte man davon abhängig machen, wie viele Nullwerte in den Fremdschlüsselattributen entstehen.
  Ist beispielsweise die Teilnahme von $E_1$ an der 1:1-Beziehung total, so ist es sinnvoll, den Primärschlüssel von $\mathcal{R}_{E_2}$ in $\mathcal{R}_{E_1}$ zu integrieren: es entstehen gar keine Nullwerte.
- Nehmen nur wenige Instanzen der beiden Entity-Typen an der Beziehung teil, sollte auf eine Zusammenfassung verzichtet werden.

# Umsetzung ER-Schema in relationales Schema

## 1:1-Beziehungstypen (Fortsetzung)

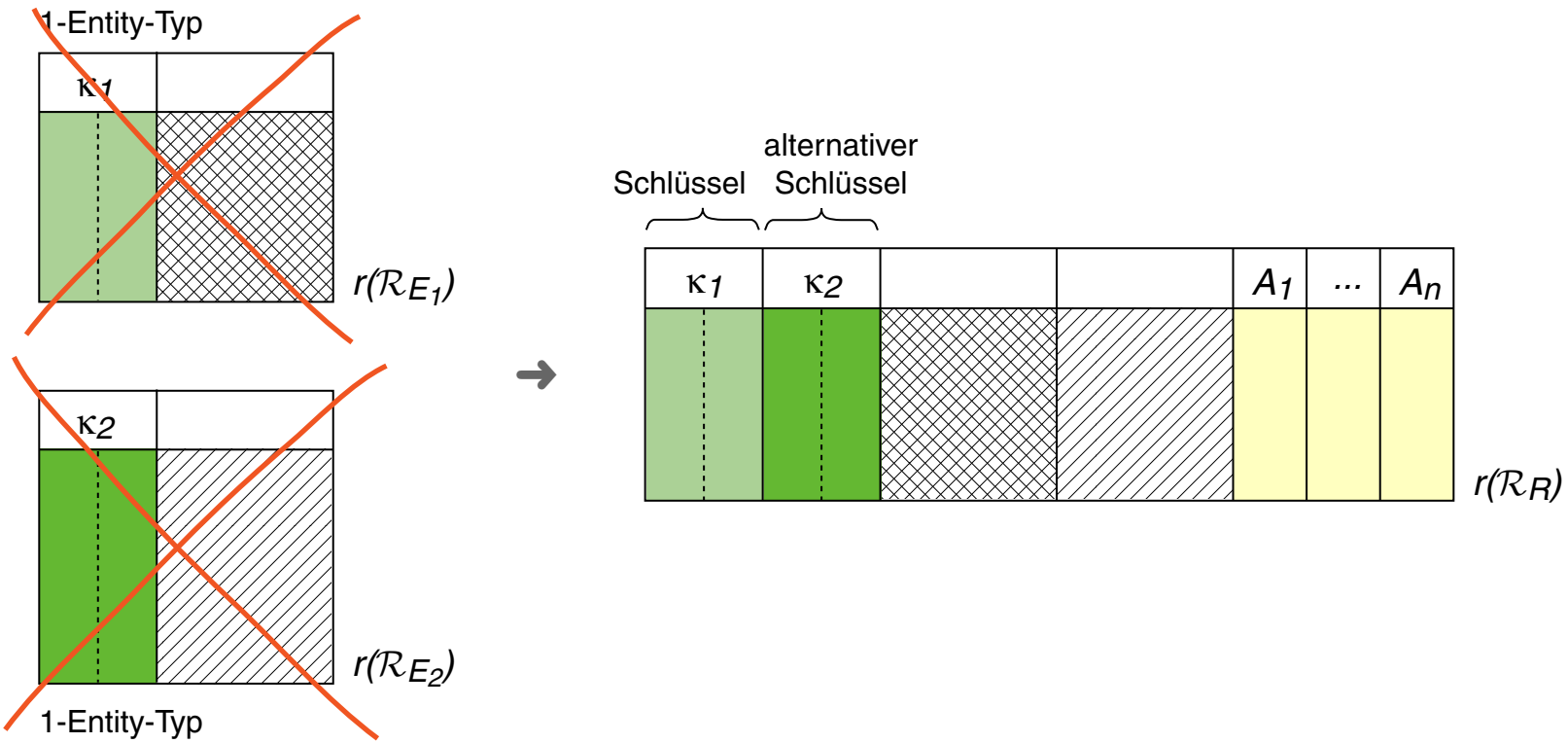
Ist die Teilnahme beider Entity-Typen am Beziehungstyp total – existiert also eine bijektive Abbildung zwischen $E_1$ und $E_2$ – lassen sich $\mathcal{R}_{E_1}$ und $\mathcal{R}_{E_2}$ in *einem* Relationenschema zusammenfassen. Merged-Relation [Elmasri/Navathe 2010]:

($\star$) Die Primärschlüssel beider Entity-Typen sind Schlüssel im zusammengefassten Relationenschema; von ihnen wird einer als Primärschlüssel gewählt. [Kapazitätserhaltung]

# Umsetzung ER-Schema in relationales Schema

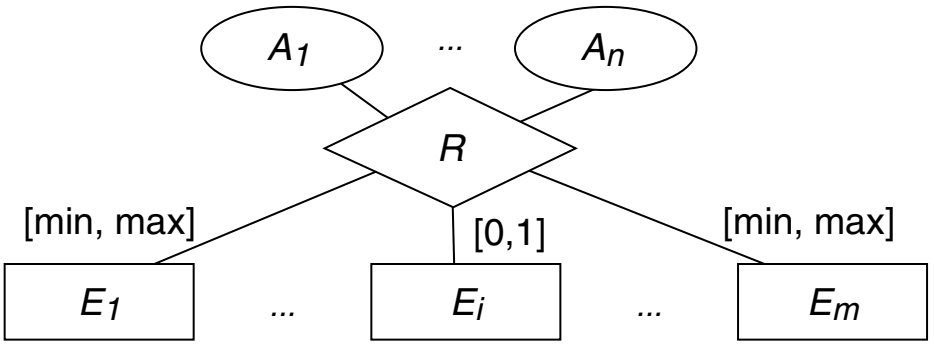
## 1:1-Beziehungstypen (Fortsetzung)

(⋆) Merged-Relation:



# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen mit $[min, max]$-Beschränkung [Sonderfall 3]

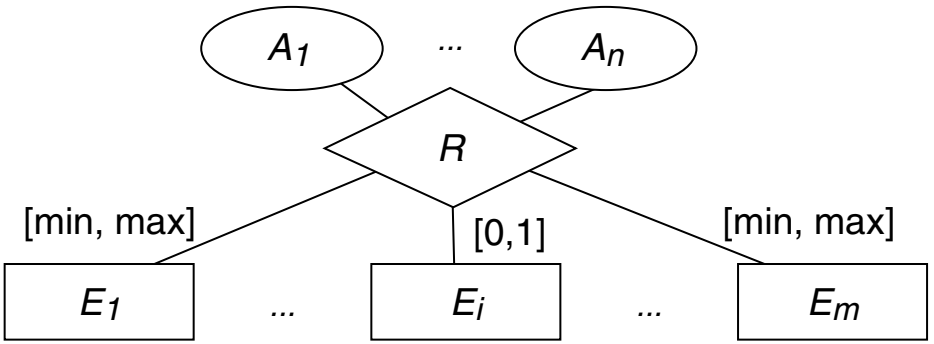



(a) $m$-äre Beziehungstypen mit $[0, 1]$-Beschränkung für Entity-Typ $E_i$:
$R(E_1[min_1, max_1], \ldots, E_i[0, 1], \ldots, E_m[min_m, max_m])$

- Der Primärschlüssel von $\mathcal{R}_{E_i}$ wird ein Schlüssel von $\mathcal{R}_R$.

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen mit $[min, max]$-Beschränkung [Sonderfall 3]




(a) $m$-äre Beziehungstypen mit $[0, 1]$-Beschränkung für Entity-Typ $E_i$:
$R(E_1[min_1, max_1], \ldots, E_i[0, 1], \ldots, E_m[min_m, max_m])$

- Der Primärschlüssel von $\mathcal{R}_{E_i}$ wird ein Schlüssel von $\mathcal{R}_R$.

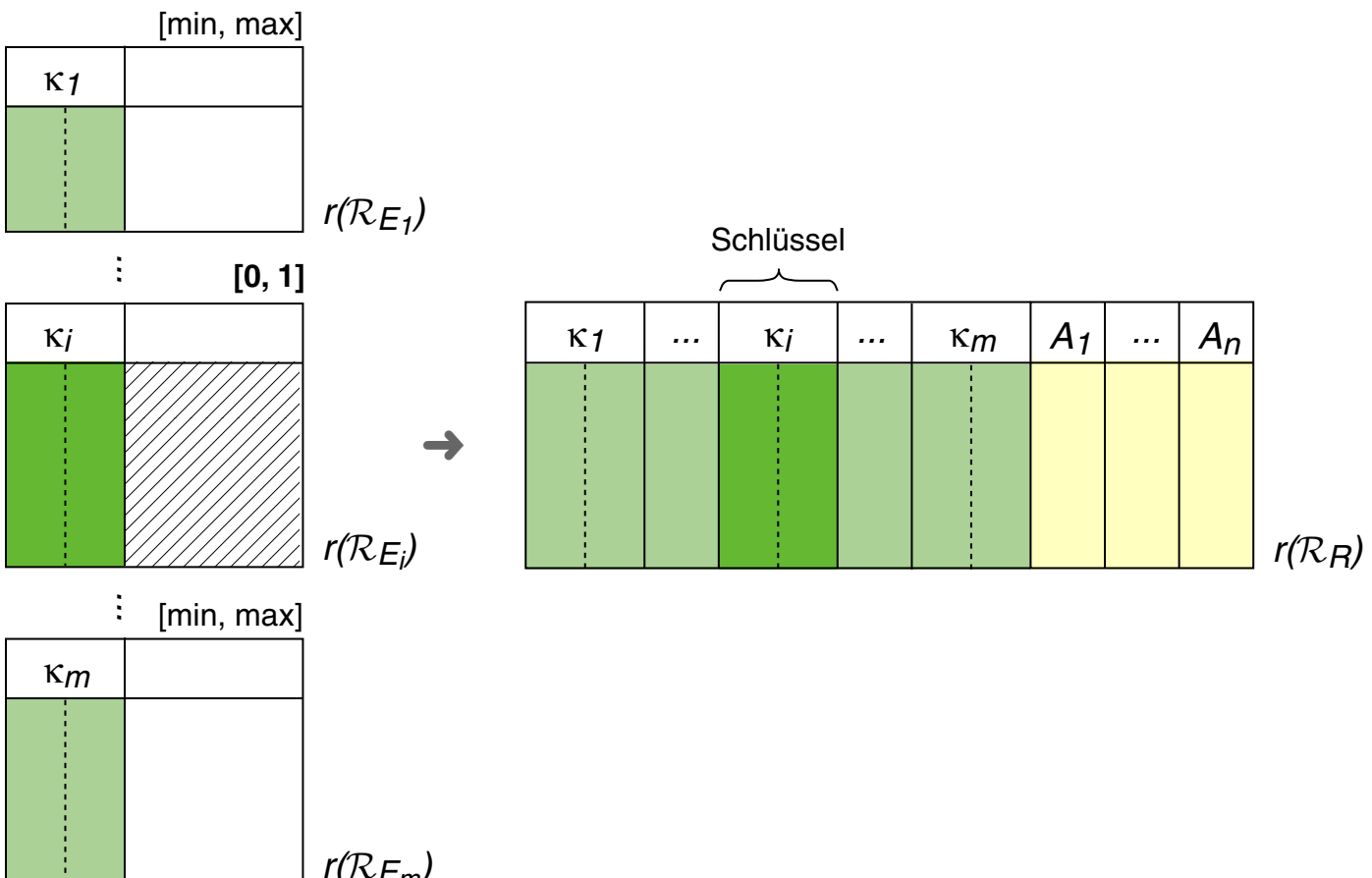
(b) $m$-äre Beziehungstypen mit $[1, 1]$-Beschränkung für Entity-Typ $E_i$:
$R(E_1[min_1, max_1], \ldots, E_i[1, 1], \ldots, E_m[min_m, max_m])$

- Der Primärschlüssel von $\mathcal{R}_{E_i}$ wird ein Schlüssel von $\mathcal{R}_R$.
- Die Relationenschemata $\mathcal{R}_R$ und $\mathcal{R}_{E_i}$ können zusammengefasst werden. Alle Schlüssel von $\mathcal{R}_{E_i}$ werden auch Schlüssel von $\mathcal{R}_R$.

# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen mit $[min, max]$-Beschränkung

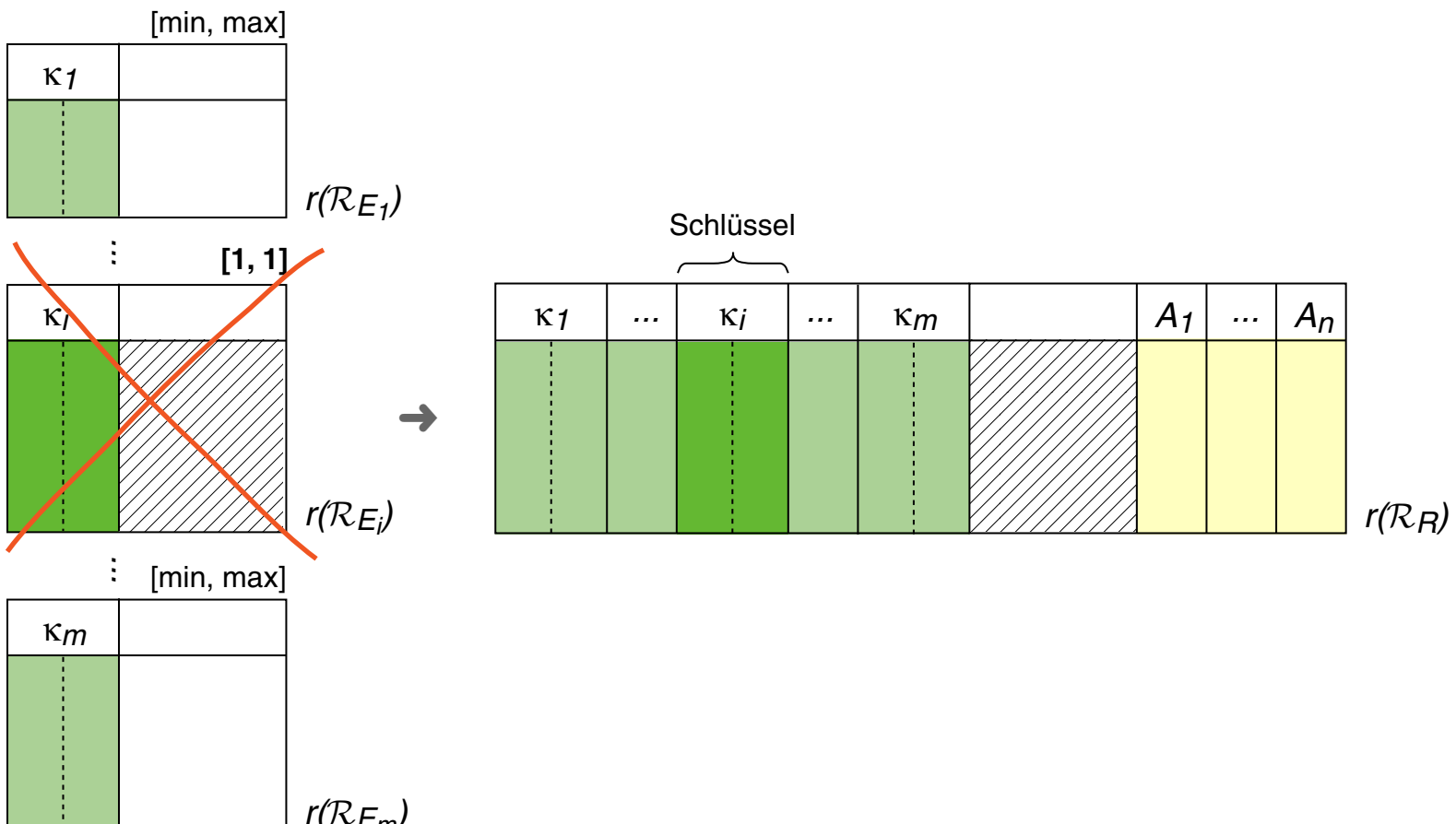
zu (a) Cross-Reference:



# Umsetzung ER-Schema in relationales Schema

## Beziehungstypen mit $[min, max]$-Beschränkung

zu (b) Zusammenfassung von $\mathcal{R}_R$ und $\mathcal{R}_{E_i}$:

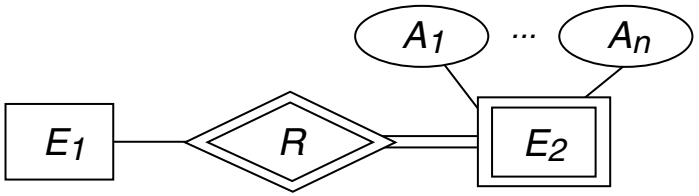


Bemerkungen:

- Eine $[0,1]$- bzw. $[1,1]$-Beschränkung qualifiziert den Schlüssel des zugehörigen Entity-Typs $E_i$ offensichtlich als Schlüssel für den Beziehungstyp $R$, denn jedes Tupel vom Typ $R$ ist höchsten bzw. genau mit einer Instanz von $E_i$ assoziiert.
- Für $m = 2$ und Vorliegen einer $[0,1]$-Beschränkung bei einem Entity-Typ entspricht die Umsetzung der Cross-Reference für binäre 1:n-Beziehungen.
- Für $m = 2$ und Vorliegen einer $[1,1]$-Beschränkung bei einem Entity-Typ entspricht die Umsetzung der Zusammenfassung für binäre 1:n-Beziehungen.
- Für $m = 2$ und Vorliegen einer $[1,1]$-Beschränkung bei *beiden* Entity-Typen ist eine Umsetzung als Merged-Relation wie bei binären 1:1-Beziehungen möglich.

# Umsetzung ER-Schema in relationales Schema

## Existenzabhängige Entity-Typen [Sonderfall 4]

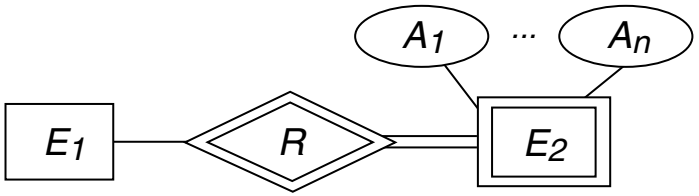



$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa_1 \cup \kappa_2$ bzw. $\{ID_1\} \cup \kappa_2$

# Umsetzung ER-Schema in relationales Schema

## Existenzabhängige Entity-Typen [Sonderfall 4]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

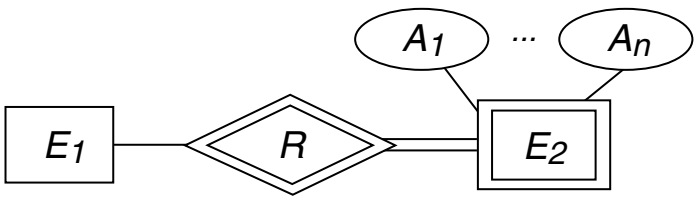
Schlüssel: $\kappa_1 \cup \kappa_2$ bzw. $\{ID_1\} \cup \kappa_2$

Umsetzung:

1. Dem abhängigen Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.

# Umsetzung ER-Schema in relationales Schema

## Existenzabhängige Entity-Typen [Sonderfall 4]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

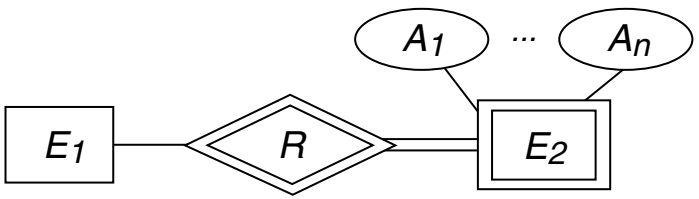
Schlüssel: $\kappa_1 \cup \kappa_2$ bzw. $\{ID_1\} \cup \kappa_2$

Umsetzung:

1. Dem abhängigen Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.
2. Die Attribute in $\kappa_1$ (bzw. $ID_1$) von $\mathcal{R}_{E_1}$ werden Attribute von $\mathcal{R}_{E_2}$ und stellen dort einen entsprechenden Fremdschlüssel dar.

# Umsetzung ER-Schema in relationales Schema

## Existenzabhängige Entity-Typen [Sonderfall 4]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa_1 \cup \kappa_2$ bzw. $\{ID_1\} \cup \kappa_2$

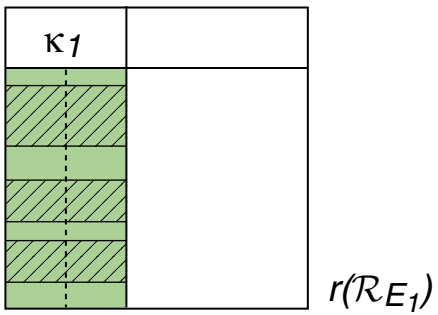
Umsetzung:

1. Dem abhängigen Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.
2. Die Attribute in $\kappa_1$ (bzw. $ID_1$) von $\mathcal{R}_{E_1}$ werden Attribute von $\mathcal{R}_{E_2}$ und stellen dort einen entsprechenden Fremdschlüssel dar.
3. Die Vereinigung des partiellen Schlüssels $\kappa_2$ von $E_2$ mit dem Primärschlüssel $\kappa_1$ (bzw. $\{ID_1\}$) von $E_1$ bildet den Schlüssel für $\mathcal{R}_{E_2}$.

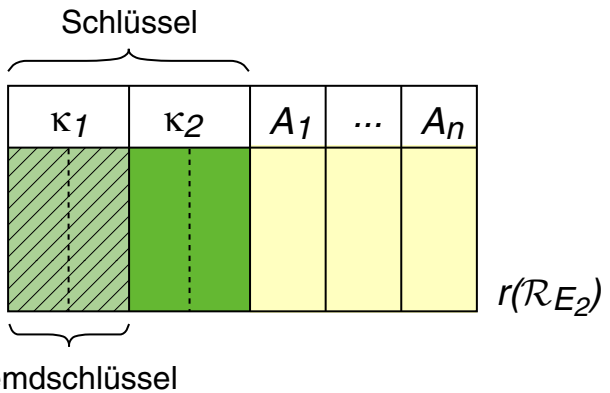
# Umsetzung ER-Schema in relationales Schema

## Existenzabhängige Entity-Typen

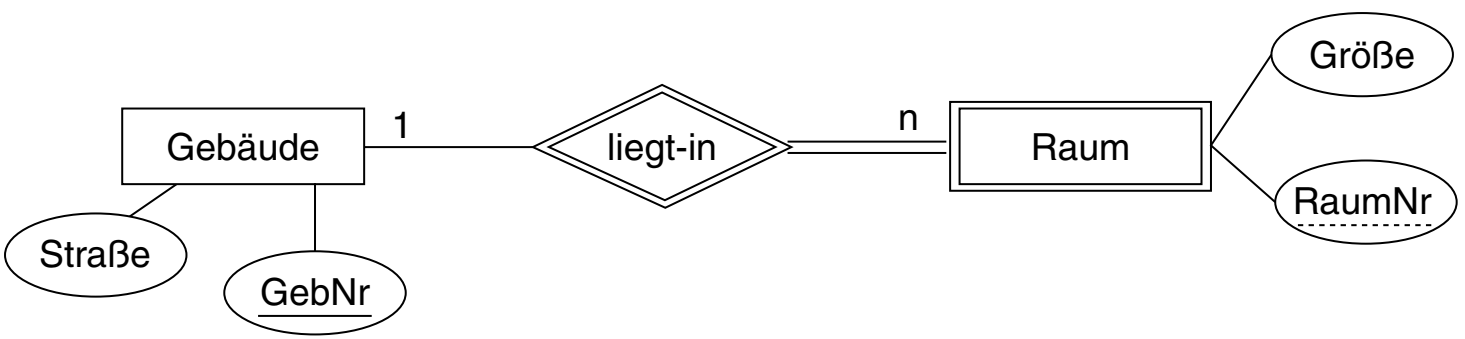
Regulärer Entity-Typ:




Abhängiger Entity-Typ:

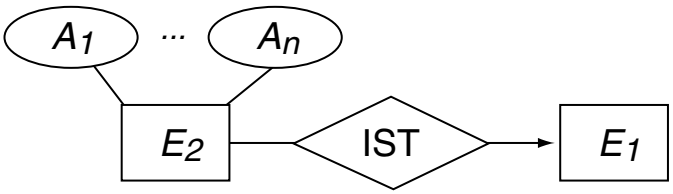



Beispiel:



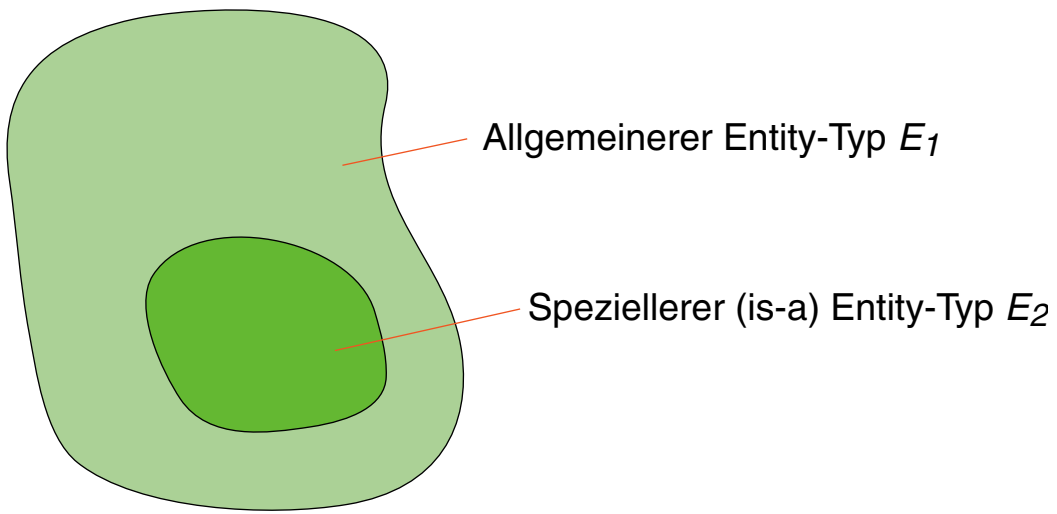
# Umsetzung ER-Schema in relationales Schema

## IST-Beziehungstypen [Sonderfall 5]



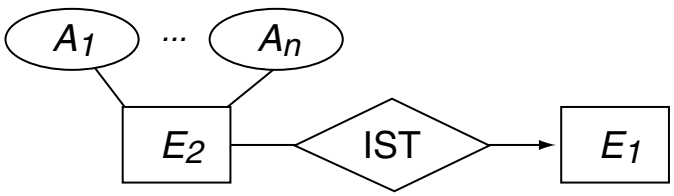

$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa_1$ bzw. $\{ID_1\}$



# Umsetzung ER-Schema in relationales Schema

## IST-Beziehungstypen [Sonderfall 5]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

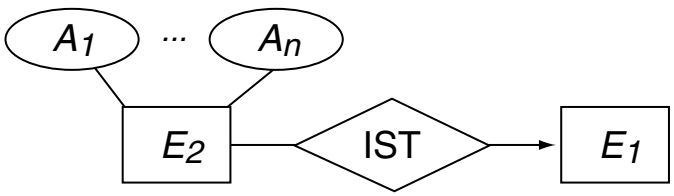
Schlüssel: $\kappa_1$ bzw. $\{ID_1\}$

Umsetzung:

1. Dem speziellerem Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.

# Umsetzung ER-Schema in relationales Schema

## IST-Beziehungstypen [Sonderfall 5]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

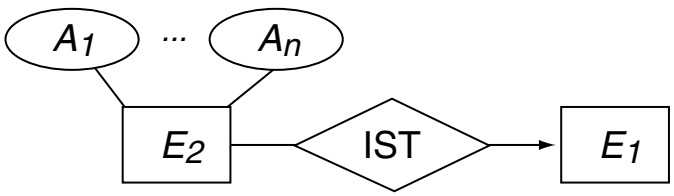
Schlüssel: $\kappa_1$ bzw. $\{ID_1\}$

Umsetzung:

1. Dem speziellerem Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.
2. Die Attribute in $\kappa_1$ (bzw. $ID_1$) von $\mathcal{R}_{E_1}$ werden Attribute von $\mathcal{R}_{E_2}$ und stellen dort einen entsprechenden „Fremdschlüssel“ dar.

# Umsetzung ER-Schema in relationales Schema

## IST-Beziehungstypen [Sonderfall 5]




$$\mathcal{R}_{E_2} = \{\underline{ID_1}, A_1, \ldots, A_n\}$$

Schlüssel: $\kappa_1$ bzw. $\{ID_1\}$

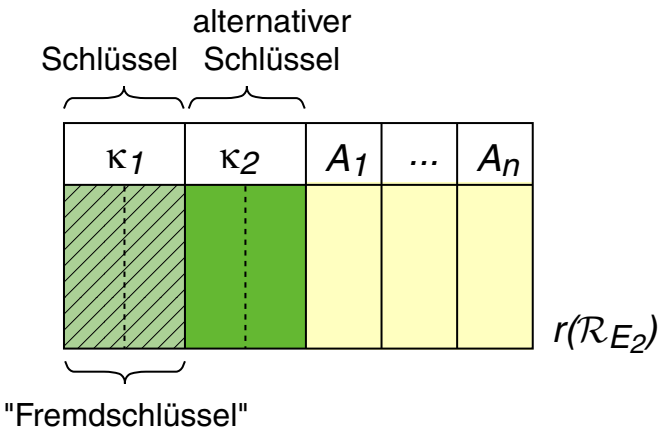
Umsetzung:

1. Dem speziellerem Entity-Typ $E_2$ wird Relationenschema $\mathcal{R}_{E_2}$ zugeordnet. Die Attribute $A_1, \ldots, A_n$ von $E_2$ werden Attribute von $\mathcal{R}_{E_2}$.
2. Die Attribute in $\kappa_1$ (bzw. $ID_1$) von $\mathcal{R}_{E_1}$ werden Attribute von $\mathcal{R}_{E_2}$ und stellen dort einen entsprechenden „Fremdschlüssel“ dar.
3. Der Primärschlüssel $\kappa_1$ (bzw. $\{ID_1\}$) von $E_1$ wird Schlüssel für $\mathcal{R}_{E_2}$.

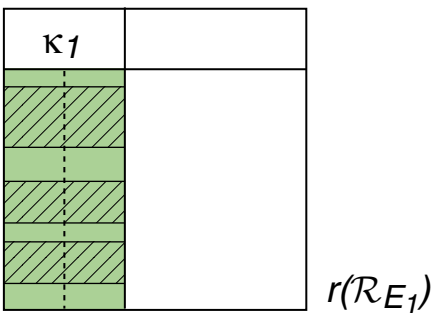
# Umsetzung ER-Schema in relationales Schema

## IST-Beziehungstypen

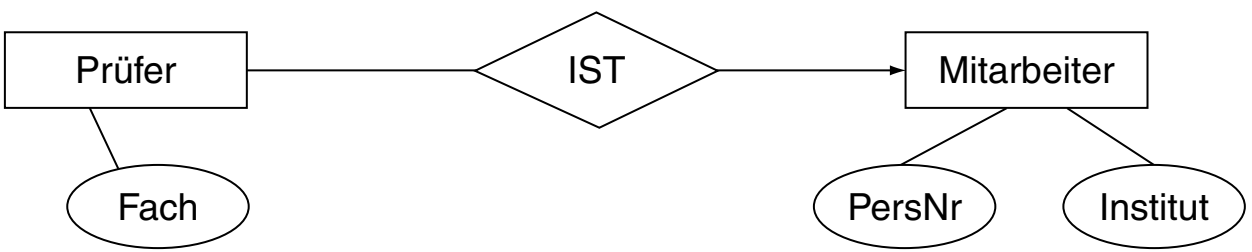
Speziellerer (is-a) Entity-Typ:




Allgemeinerer Entity-Typ:




Beispiel:



Bemerkungen:

- Es wird die Bezeichnung „Fremdschlüssel“ benutzt, obwohl es sich bei der Spezialisierung nicht um einen Verweis auf einen anderen Entity-Typ handelt, sondern um eine Rollenbeschreibung für ein und denselben Entity-Typ.
- Ein spezialisierter Entity-Typ $E_2$ kann bereits einen Schlüssel $\kappa_2$ unabhängig von dem Entity-Typ $E_1$ besitzen, von dem er spezialisiert ist. In diesem Fall hat man für $E_2$ die Wahl zwischen zwei Schlüsseln, von denen einer als Primärschlüssel festzulegen ist.
- Bei mehrstufigen IST-Beziehungstypen wird der Primärschlüssel – und damit die Identität – top-down (vom allgemeineren zum spezielleren Entity-Typ) vererbt. Damit ist auch eine Transformationsreihenfolge vorgegeben.

# Umsetzung ER-Schema in relationales Schema

## Reihenfolge der Regelanwendung [Elmasri/Navathe 2010]

1. Transformation der regulären Entity-Typen.
2. Transformation der abhängigen Entity-Typen.
3. Transformation der 1:1-Beziehungstypen.
4. Transformation der 1:n-Beziehungstypen.
5. Transformation der n:m-Beziehungstypen.
6. Transformation der übrigen Beziehungstypen.
7. Transformation der IST-Beziehungstypen.

# Umsetzung ER-Schema in relationales Schema

## Zusammenfassung wichtiger Regeln

| Konzept im ER-Modell | Konzept im relationalen Modell |
|---|---|
| Entity-Typ $E$ | Relationenschema $\mathcal{R}_E$ |
| Attribute $A_1, \ldots, A_n$ von $E$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_E$ |
| Primärschlüssel $\kappa \subseteq \{A_1, \ldots, A_n\}$ von $E$ | Primärschlüssel $\kappa$ von $\mathcal{R}_E$ |
| Beziehungstyp $R(E_1, ..., E_m; A_1, ..., A_n)$ | Relationenschema $\mathcal{R}_R$ |
| Attribute $A_1, \ldots, A_n$ von $R$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_R$ |
| Attribute in den Primärschlüsseln $\kappa_i$ der $E_i$ | Attribute von $\mathcal{R}_R$ (als Fremdschlüssel) |
| 1:n-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| 1:1-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1$ und $\kappa_2$ werden jeweils Schlüssel von $\mathcal{R}_R$, $\kappa_1$ oder $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| n:m-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_R$ |
| $E_2$ hängt ab von $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_{E_2}$ |
| IST-Beziehungstyp: $E_2$ IST $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1$ wird Schlüssel von $\mathcal{R}_{E_2}$ |

# Umsetzung ER-Schema in relationales Schema

## Zusammenfassung wichtiger Regeln

| **Konzept im ER-Modell** | **Konzept im relationalen Modell** |
|---|---|
| Entity-Typ $E$ | Relationenschema $\mathcal{R}_E$ |
| Attribute $A_1, \ldots, A_n$ von $E$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_E$ |
| Primärschlüssel $\kappa \subseteq \{A_1, \ldots, A_n\}$ von $E$ | Primärschlüssel $\kappa$ von $\mathcal{R}_E$ |
| Beziehungstyp $R(E_1, ..., E_m; A_1, ..., A_n)$ | Relationenschema $\mathcal{R}_R$ |
| Attribute $A_1, \ldots, A_n$ von $R$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_R$ |
| Attribute in den Primärschlüsseln $\kappa_i$ der $E_i$ | Attribute von $\mathcal{R}_R$ (als Fremdschlüssel) |
| 1:n-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| 1:1-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1$ und $\kappa_2$ werden jeweils Schlüssel von $\mathcal{R}_R$, $\kappa_1$ oder $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| n:m-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_R$ |
| $E_2$ hängt ab von $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_{E_2}$ |
| IST-Beziehungstyp: $E_2$ IST $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1$ wird Schlüssel von $\mathcal{R}_{E_2}$ |

# Umsetzung ER-Schema in relationales Schema

## Zusammenfassung wichtiger Regeln

| **Konzept im ER-Modell** | **Konzept im relationalen Modell** |
|---|---|
| Entity-Typ $E$ | Relationenschema $\mathcal{R}_E$ |
| Attribute $A_1, \ldots, A_n$ von $E$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_E$ |
| Primärschlüssel $\kappa \subseteq \{A_1, \ldots, A_n\}$ von $E$ | Primärschlüssel $\kappa$ von $\mathcal{R}_E$ |
| Beziehungstyp $R(E_1, ..., E_m; A_1, ..., A_n)$ | Relationenschema $\mathcal{R}_R$ |
| Attribute $A_1, \ldots, A_n$ von $R$ | Attribute $A_1, \ldots, A_n$ von $\mathcal{R}_R$ |
| Attribute in den Primärschlüsseln $\kappa_i$ der $E_i$ | Attribute von $\mathcal{R}_R$ (als Fremdschlüssel) |
| 1:n-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| 1:1-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1$ und $\kappa_2$ werden jeweils Schlüssel von $\mathcal{R}_R$, $\kappa_1$ oder $\kappa_2$ wird Primärschlüssel von $\mathcal{R}_R$ |
| n:m-Beziehungstyp zwischen $E_1$ und $E_2$ | $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_R$ |
| $E_2$ hängt ab von $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1 \cup \kappa_2$ wird Schlüssel von $\mathcal{R}_{E_2}$ |
| IST-Beziehungstyp: $E_2$ IST $E_1$ | $\mathcal{R}_{E_2}$ erhält auch alle Attribute in $\kappa_1$, $\kappa_1$ wird Schlüssel von $\mathcal{R}_{E_2}$ |