Datenbanken

Das Relationale Datenmodell

Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Dennis Heinrich (Übungen)

und studentische Tutoren

RDM: Überblick über die Konzepte (1)

- Eine Datenbank ist eine Menge benannter **Relationen**
- Eine Relation ist eine Menge von Elementen (**Tupeln**)
  - deren Struktur durch **Attribute** definiert,
  - deren Identität durch **Schlüssel** realisiert und
  - deren Werte durch **Domänen** kontrolliert werden

- Relationen werden meist durch **Tabellen** dargestellt, wobei jede Tabelle aus Zeilen und Spalten besteht
- Jede Zeile repräsentiert ein Element der Relation und wird auch als **Tupel** bezeichnet
- Die Zahl der Zeilen ist variabel und wird **Kardinalität** der Relation genannt
- Die Spalten der Tabellen enthalten die **Attribute** der Relation

RDM: Überblick über die Konzepte (2)

- Die Zahl der **Spalten** einer Tabelle wird im Schema festgelegt
- Jeder Spalte ist eine **Domäne** zugeordnet, welche die zulässigen Werte für das Attribut in allen Zeilen festlegt
- Jede Tabelle besitzt einen **Primärschlüssel**, der ein einzelnes Attribut oder eine Kombination von Attributen ist, so dass eine eindeutige Identifikation jedes Tupels innerhalb der Tabelle ermöglicht wird

- **Beziehungen** zwischen Datenobjekten werden durch Identifikation des referenzierten Objektes über seinen Primärschlüssel repräsentiert (➠ **assoziative Identifikation**)
- Einen Schlüssel, der in Relation A zur Identifikation eines Tupels in Relation B benutzt wird, bezeichnet man als **Fremdschlüssel**

RDM: Projektdatenbank

Nr	*Titel*	*Budget*
100	DB Fahrpläne	300.000
200	ADAC Kundenstamm	100.000
300	Telekom Statistik	200.000

Projekte

Nr	*Kurz*
100	MFSW
100	UXSW
100	LTSW
200	UXSW
200	PERS
300	MFSW

Projektdurchführung

Kurz	*Name*	*Oberabt*
MFSW	Mainframe SW	LTSW
UXSW	Unix SW	LTSW
PCSW	PC SW	LTSW
LTSW	Leitung SW	**NULL**
PERS	Personal	**NULL**

Abteilungen

Projektdatenbank

RDM: Tabellen und Schlüssel (1)

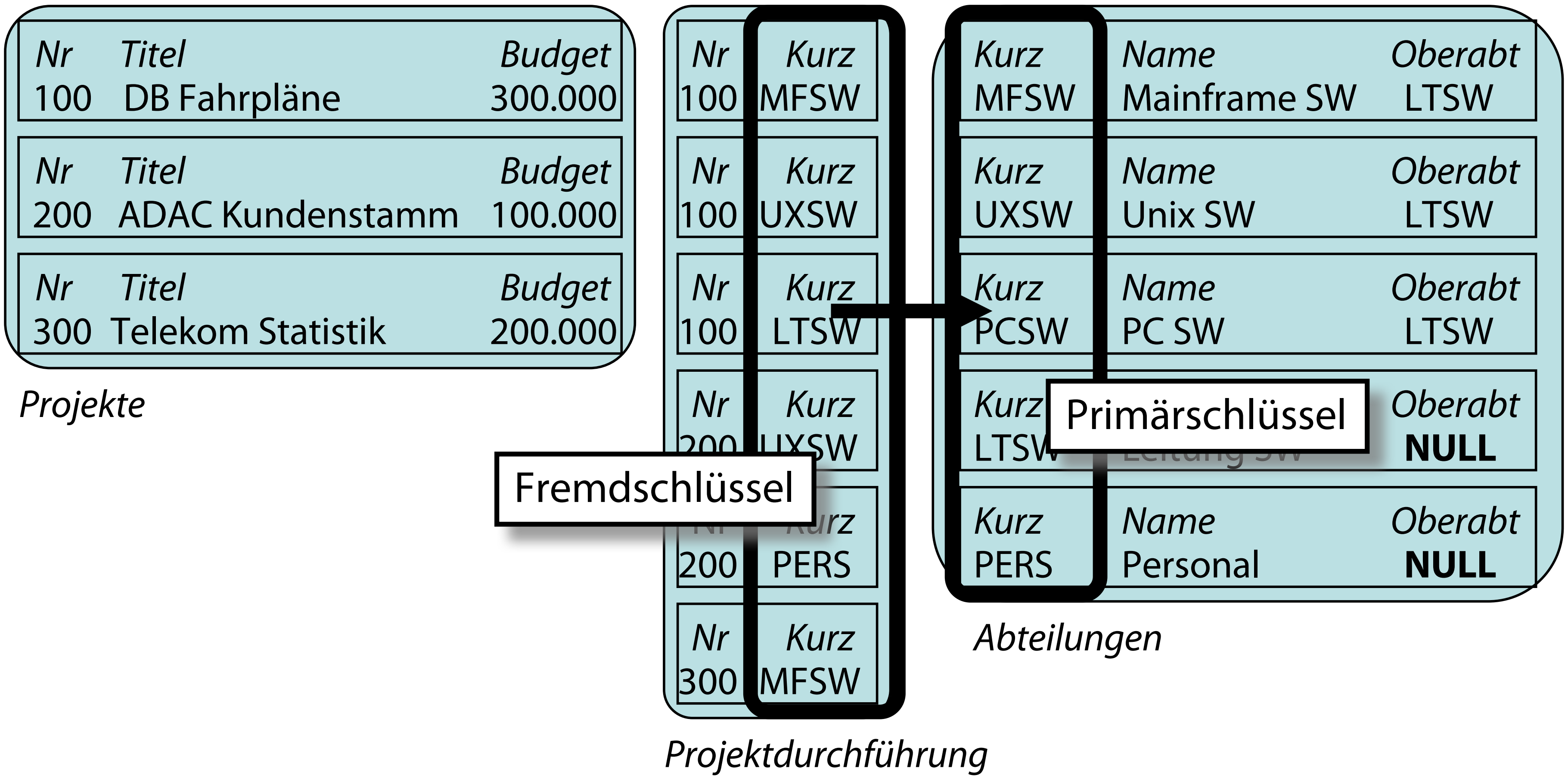
Duplikate bzgl. der Schlüsselwerte sind nicht erlaubt, d.h. die Gesamtheit aller Attribute bildet automatisch einen Schlüsselkandidaten

Oft ist jedoch die Einführung eines künstlichen Schlüssels z.B. einer eindeutigen Nummer (ID) sinnvoll

Eine Relation mit Primärschlüssel repräsentiert eine Funktion von den Primärschlüsselattributen zu den Nicht-Schlüsselattributen

Beispiel: Kurz ➠ (Name, Oberabt), Kurz ➠ Name, Kurz ➠ Oberabt

RDM: Tabellen und Schlüssel (2)



Projektdatenbank

RDM: Datendefinition

Schemadefinition der Projektdatenbank:

```
create table Projekte
( Nr integer not null,
  Titel char(30) not null,
  Budget decimal(10,2) not null,
  primary key(Nr) );
```

```
create table
Projektdurchfuehrung
( Nr integer not null,
  Kurz char(4) not null,
  primary key(Nr, Kurz) );
```

```
create table Abteilungen
( Kurz char(4) not null,
  Name char(30) not null,
  Oberabt char(4),
  primary key(Kurz) );
```

Referentielle Integrität in SQL: Kapitel 3.2

RDM: Referentielle Integrität

Referentielle Integrität: Zu jedem benutzten Fremdschlüssel existiert ein Tupel mit einem entsprechenden Primärschlüsselwert in der referenzierten Tabelle.

Überprüfung der referentiellen Integrität ist notwendig beim

- Einfügen eines neuen Fremdschlüsselwertes in eine Beziehungstabelle. Das referenzierte Objekt mit diesem Wert als Primärschlüssel muss existieren.
- Löschen eines Tupels aus einer Entitätentabelle. Auf dieses Tupel dürfen keine Referenzen bestehen. Gibt es noch Referenzen, bieten sich mehrere Möglichkeiten an:
  - Eine Fehlermeldung wird erzeugt.
  - Propagierung der Löschoperation, das referenzierende Tupel wird ebenfalls gelöscht (➠ *kaskadiertes Löschen*).
  - Die Referenzen können durch Setzen des Fremdschlüssels auf einen Nullwert ungültig gemacht werden, sofern dieser nicht Bestandteil des Schlüssels ist.

Referentielle Integrität

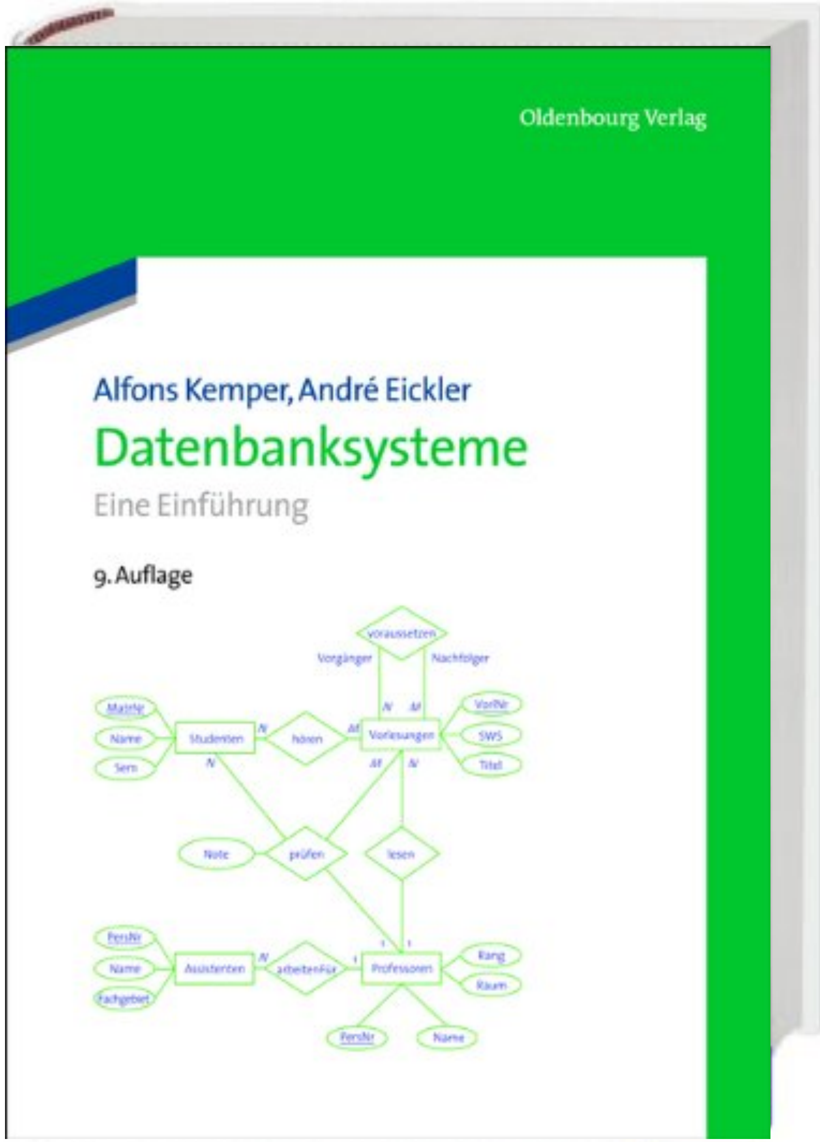
Im allgemeinen besteht ein Fremdschlüssel einer Tabelle `T` aus einer Liste von Spalten, der eine typkompatible Liste von Spalten in `S` entspricht:

```
create table T
  (...
   constraint Name
     foreign key(A1, A2, ..., An) references (S(B1, B2, ..., Bn))
```

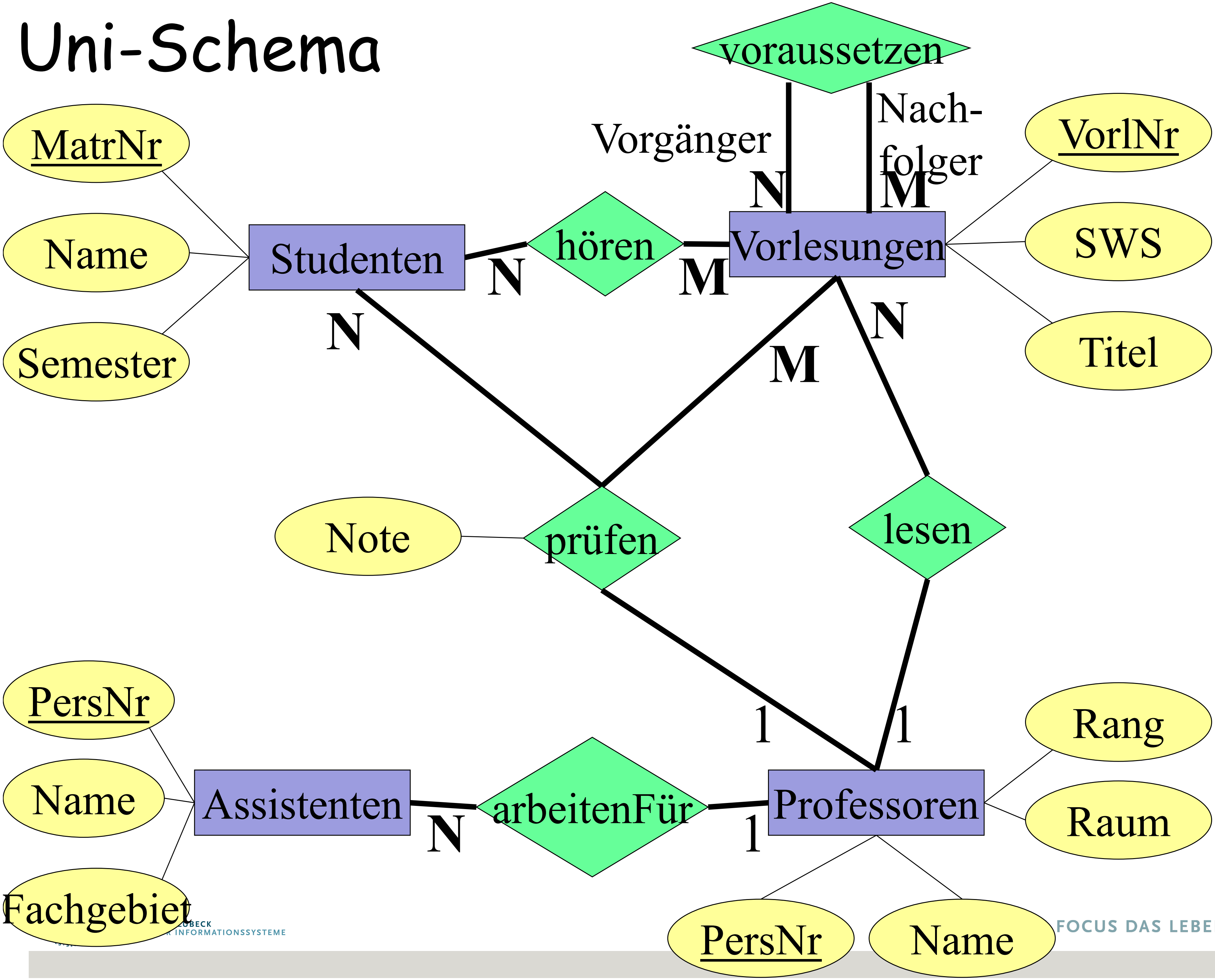
Sind $B_1, B_2, \ldots, B_n$ die Primärschlüsselspalten von `S`, kann ihre Angabe entfallen.

Beachte: Rekursive Beziehungen (z.B. Abteilung : Oberabteilung) führen zu reflexiven Fremdschlüsseldeklarationen (`S` = `T`).

Acknowledgments / Skript zur Vorlesung



Uni-Schema



voraussetzen
Vorgänger
Nach-
folger
N
M
MatrNr
Name
Semester
Studenten
hören
Vorlesungen
VorlNr
SWS
Titel
N
M
N
M
N
Note
prüfen
lesen
1
1
PersNr
Name
Fachgebiet
Assistenten
arbeitenFür
Professoren
Rang
Raum
N
1
PersNr
Name

Relationale Darstellung von Entitytypen

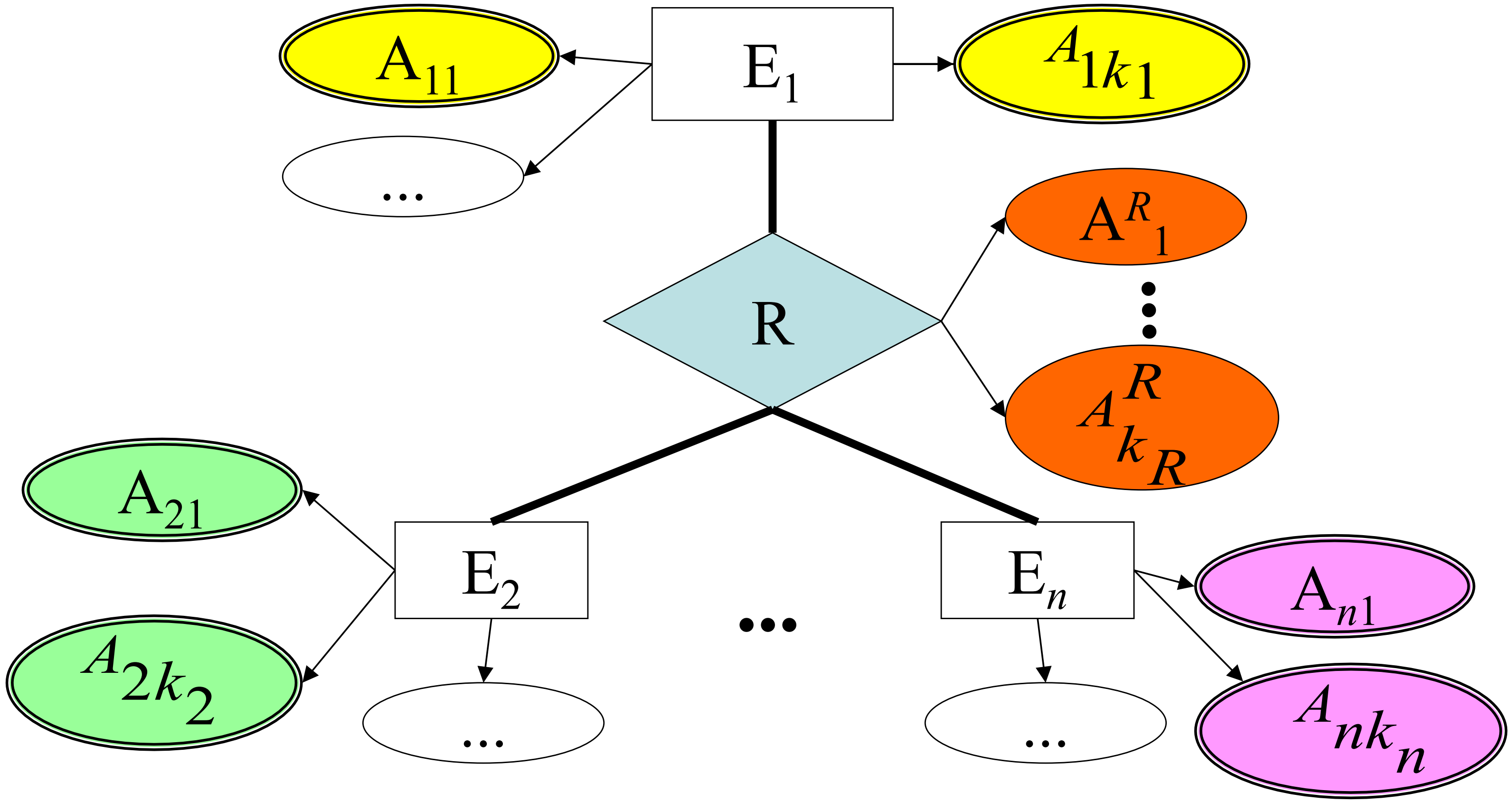
Studenten: {[<u>MatrNr:integer</u>, *Name*: *string, Semester*: *integer*]}

Vorlesungen: {[<u>VorlNr:integer</u>, *Titel*: *string, SWS*: *integer*]}

Professoren: {[<u>PersNr:integer</u>, *Name*: *string, Rang*: *string, Raum*: *integer*]}

Assistenten: {[<u>PersNr:integer</u>, *Name*: *string, Fachgebiet*: *string*]}

Relationale Darstellung von Beziehungen



$$R: \{[\underbrace{A_{11},\ldots,A_{1k_1}}_{\text{Schlüssel von } E_1}, \underbrace{A_{21},\ldots,A_{2k_2}}_{\text{Schlüssel von } E_2}, \ldots, \underbrace{A_{n1},\ldots,A_{nk_n}}_{\text{Schlüssel von } E_n}, \underbrace{A^R_1,\ldots,A^R_{k_R}}_{\text{Attribute von } R}]\}$$

Beziehungen unseres Beispiel-Schemas

hören : {[MatrNr: integer, VorlNr: integer]}

lesen : {[PersNr: integer, VorlNr: integer]}

arbeitenFür : {[AssistentenPersNr: integer, *ProfPersNr: integer*]}

voraussetzen : {[Vorgänger: integer, Nachfolger: integer]}

prüfen : {[MatrNr: integer, VorlNr: integer, PersNr: integer,
Note: decimal]}

Schlüssel der Relationen

hören : {[<u>MatrNr: integer, VorlNr: integer</u>]}

lesen : {[PersNr: integer, <u>VorlNr: integer</u>]}

Warum nicht beide Attribute?

arbeitenFür : {[<u>AssistentenPersNr: integer</u>, *ProfPersNr: integer*]}

voraussetzen : {[<u>Vorgänger: integer, Nachfolger: integer</u>]}

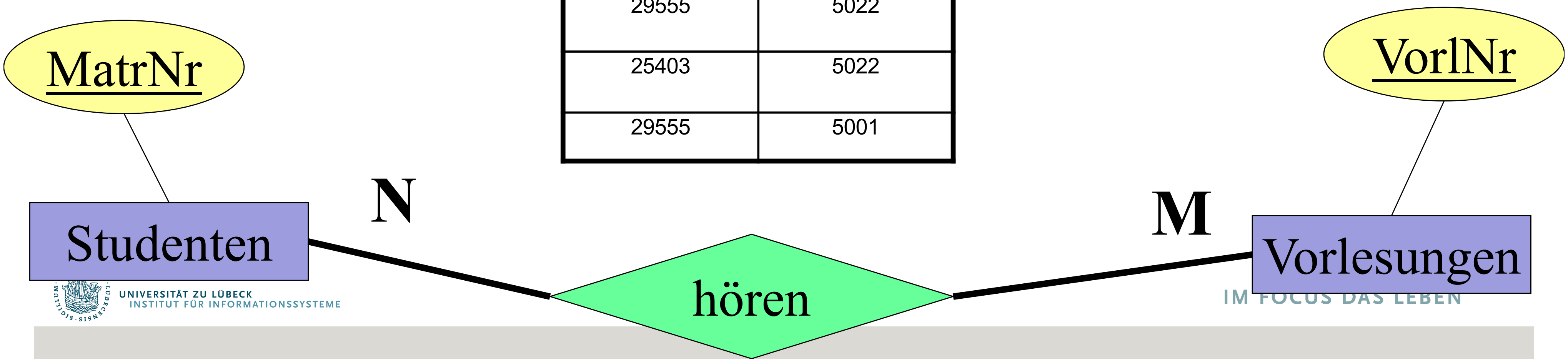
prüfen : {[<u>MatrNr: integer, VorlNr: integer</u>, PersNr: integer,
Note: decimal]}

Ausprägung der Beziehung *hören*

Studenten	
MatrNr	...
26120	...
27550	...
...	...

hören	
MatrNr	VorlNr
26120	5001
27550	5001
27550	4052
28106	5041
28106	5052
28106	5216
28106	5259
29120	5001
29120	5041
29120	5049
29555	5022
25403	5022
29555	5001

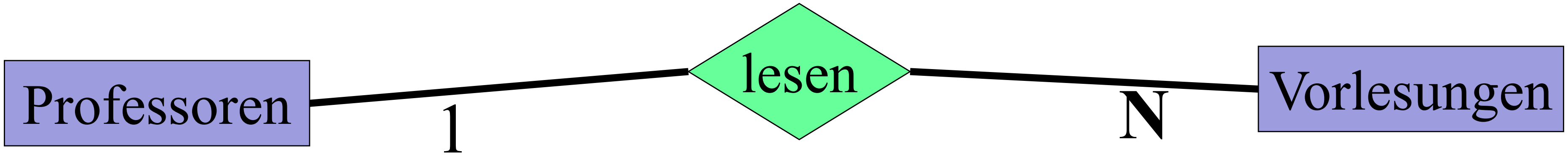
Vorlesungen	
VorlNr	...
5001	...
4052	...
...	...



Notation für Relationenschemata

- Schema: Tabellenname = {[Attr1: Typ1, Attr2: Typ2, ...]}
- In eckigen Klammern [...] wird angegeben,
  wie die Tupel aufgebaut sind
- Die Mengenklammern sollen ausdrücken, dass es sich bei einer Relationenausprägung um eine Menge von Tupeln handelt
- Manchmal werden die Attribute auch als Menge benötigt:
  Wir schreiben für das Schema der Tabelle $\mathcal{R}$: $\mathcal{R}$ = {Attr1, Attr2, ...}
- Eine konkrete Relation R ist eine Teilmenge des Kreuzproduktes von dom(Attr1) x dom(Attr2) x ...

Verfeinerung des relationalen Schemas



1:N-Beziehung

Initial-Entwurf

Vorlesungen : {[*<u>VorlNr</u>, Titel, SWS*]}

Professoren : {[*<u>PersNr</u>, Name, Rang, Raum*]}

lesen: {[*<u>VorlNr</u>, PersNr*]}

Verfeinerung des relationalen Schemas

1:N-Beziehung

Initial-Entwurf

Vorlesungen : {[*<u>VorlNr</u>, Titel, SWS*]}

Professoren : {[*<u>PersNr</u>, Name, Rang, Raum*]}

lesen: {[*<u>VorlNr</u>, PersNr*]}

Verfeinerung durch Zusammenfassung

Vorlesungen : {[*<u>VorlNr</u>, Titel, SWS,* ***gelesenVon***]}

Professoren : {[*<u>PersNr</u>, Name, Rang, Raum*]}

Regel

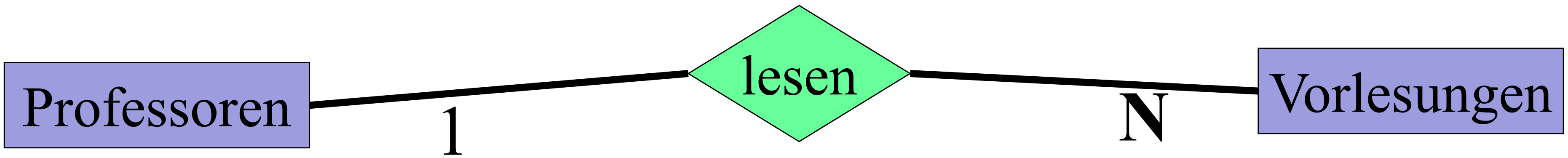
Relationen mit gleichem Schlüssel kann man zusammenfassen

aber nur diese und keine anderen!

Ausprägung von *Professoren* und *Vorlesung*

Professoren			
PersNr	Name	Rang	Raum
2125	Sokrates	C4	226
2126	Russel	C4	232
2127	Kopernikus	C3	310
2133	Popper	C3	52
2134	Augustinus	C3	309
2136	Curie	C4	36
2137	Kant	C4	7

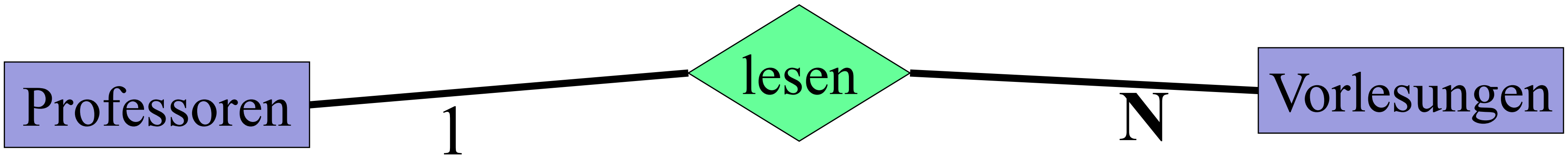
Vorlesungen			
VorlNr	Titel	SWS	Gelesen Von
5001	Grundzüge	4	2137
5041	Ethik	4	2125
5043	Erkenntnistheorie	3	2126
5049	Mäeutik	2	2125
4052	Logik	4	2125
5052	Wissenschaftstheorie	3	2126
5216	Bioethik	2	2126
5259	Der Wiener Kreis	2	2133
5022	Glaube und Wissen	2	2134
4630	Die 3 Kritiken	4	2137



Vorsicht: So geht es NICHT

Professoren				
PersNr	Name	Rang	Raum	liest
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...	...	...	...	...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

Vorlesungen		
VorlNr	Titel	SWS
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
5216	Bioethik	2
5259	Der Wiener Kreis	2
5022	Glaube und Wissen	2
4630	Die 3 Kritiken	4



Anomalien

Vorlesungen		
VorlNr	Titel	SWS
5001	Grundzüge	4
5041	Ethik	4
5043	Erkenntnistheorie	3
5049	Mäeutik	2
4052	Logik	4
5052	Wissenschaftstheorie	3
5216	Bioethik	2
5259	Der Wiener Kreis	2
5022	Glaube und Wissen	2
4630	Die 3 Kritiken	4

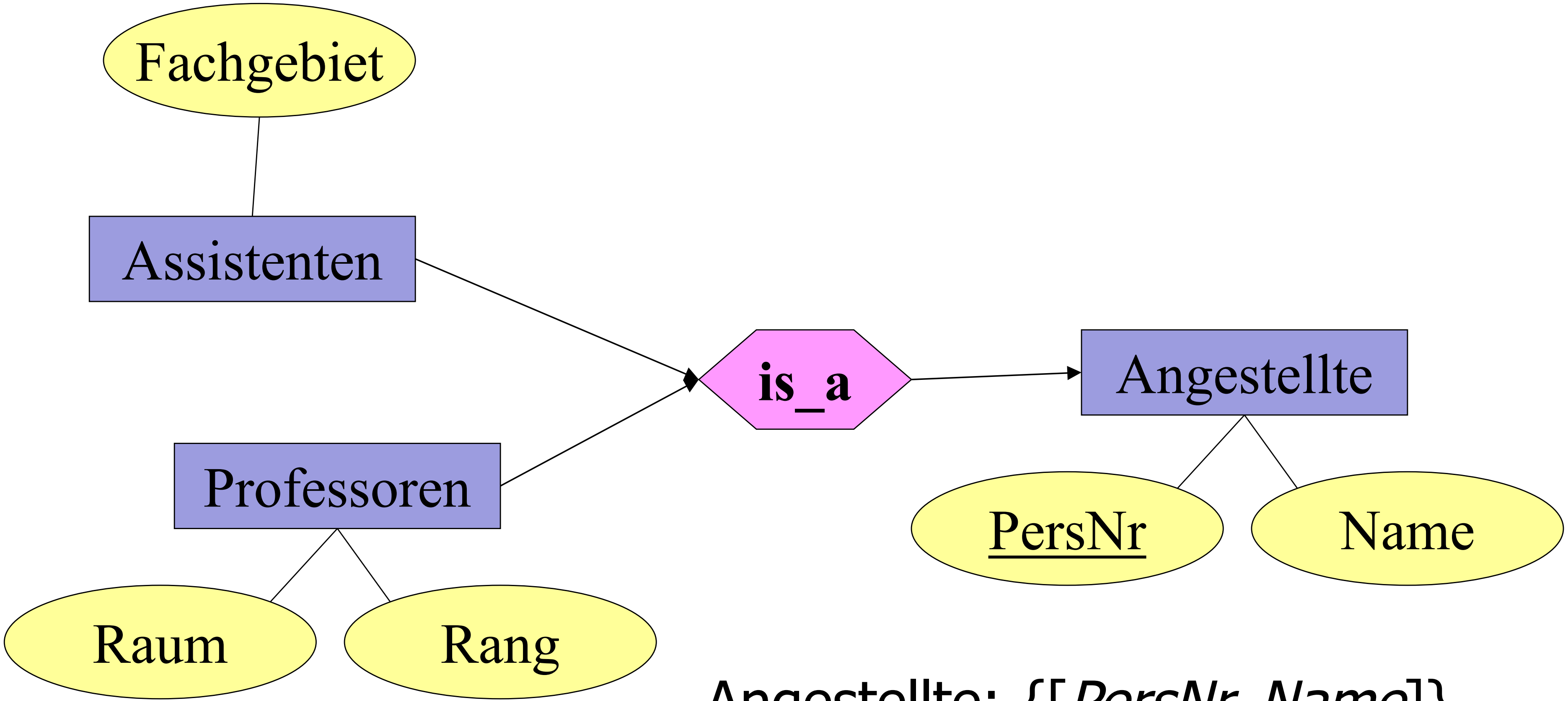
Professoren				
PersNr	Name	Rang	Raum	liest
2125	Sokrates	C4	226	5041
2125	Sokrates	C4	226	5049
2125	Sokrates	C4	226	4052
...	...	...	...	...
2134	Augustinus	C3	309	5022
2136	Curie	C4	36	??

Update-Anomalie: Was passiert, wenn Sokrates umzieht

Lösch-Anomalie: Was passiert, wenn „Glaube und Wissen“ wegfällt

Einfügeanomalie: Curie ist neu und liest noch keine Vorlesungen

Relationale Modellierung der Generalisierung



Angestellte: {[*<u>PersNr</u>*, *Name*]}

Professoren: {[*<u>PersNr</u>*, *Rang*, *Raum*]}

Assistenten: {[*<u>PersNr</u>*, *Fachgebiet*]}

Vereinbarung zur Notation

Sei $\mathcal{R}$ = *{A, B, C, D}* ein Relationenschema.

Seien r und t Tupel aus einer konkreten Relation R gemäß dem Schema $\mathcal{R}$.

Sei weiterhin $\alpha \subseteq \mathcal{R}$.

Wir vereinbaren:

r.α = t.α soll heißen, dass für alle A aus α gilt: r.A = t.A.

Funktionale Abhängigkeiten

Schema

$\mathcal{R} = \{A, B, C, D\}$

Ausprägung R

Seien $\alpha \subseteq \mathcal{R}, \ \beta \subseteq \mathcal{R}$

$\alpha \longrightarrow \beta$ genau dann wenn $\forall r, s \in R$ mit $r.\alpha = s.\alpha \Rightarrow r.\beta = s.\beta$

R			
A	*B*	*C*	*D*
a4	b2	c4	d3
a1	b1	c1	d1
a1	b1	c1	d2
a2	b2	c3	d2
a3	b2	c4	d3

$\{A\} \longrightarrow \{B\}$

$\{C, D\} \longrightarrow \{B\}$

Nicht: $\{B\} \longrightarrow \{C\}$

Notationskonvention:

$CD \longrightarrow B$

Beispiel

Stammbaum				
Kind	Vater	Mutter	Opa	Oma
Sofie	Alfons	Sabine	Lothar	Linde
Sofie	Alfons	Sabine	Hubert	Lisa
Niklas	Alfons	Sabine	Lothar	Linde
Niklas	Alfons	Sabine	Hubert	Lisa
...	...	...	Lothar	Martha
…	…	…	…	…

$$\text{Kind} \longrightarrow \text{Vater,Mutter}$$

$$\text{Kind,Opa} \longrightarrow \text{Oma}$$

$$\text{Kind,Oma} \longrightarrow \text{Opa}$$

Schlüssel

$\alpha \subseteq \mathcal{R}$ ist ein **Super-Schlüssel**, falls Folgendes gilt:

$\alpha \longrightarrow \mathcal{R}$

Wir nennen α Super-Schlüssel, weil noch nichts darüber ausgesagt wird, dass der Schlüssel α minimal ist.

β ist **voll funktional abhängig** von α genau dann wenn gilt

$\alpha \longrightarrow \beta$ und

α kann nicht mehr verkleinert werden, d.h.

$\forall A \in \alpha: (\alpha \setminus \{A\}) \longrightarrow \beta$ gilt nicht

Notation für volle funktionale Abhängigkeit: $\alpha \overset{\bullet}{\longrightarrow} \beta$

$\alpha \subseteq \mathcal{R}$ ist ein **Kandidaten-Schlüssel**, falls Folgendes gilt: $\alpha \overset{\bullet}{\longrightarrow} \mathcal{R}$

Schlüsselbestimmung

Städte			
Name	BLand	Vorwahl	EW
Frankfurt	Hessen	069	650000
Frankfurt	Brandenburg	0335	84000
München	Bayern	089	1200000
Passau	Bayern	0851	50000
...	...	...	...

Kandidaten-schlüssel von *Städte*:

- {Name,BLand}
- {Name,Vorwahl}

Beachte, dass 2 kleinere Städte dieselbe Vorwahl haben können

Bestimmung funktionaler Abhängigkeiten

Professoren: {[PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, Bland, EW, Landesregierung]}

- {PersNr} ⟶ {PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, Bland, EW, Landesregierung}
- {Ort,BLand} ⟶ {EW, Vorwahl}
- {PLZ} ⟶ {Bland, Ort, EW}
- {Bland, Ort, Straße} ⟶ {PLZ}
- {Bland} ⟶ {Landesregierung}
- {Raum} ⟶ {PersNr}

Zusätzliche Abhängigkeiten, die aus obigen abgeleitet werden können:

- {Raum} ⟶ {PersNr, Name, Rang, Raum, Ort, Straße, PLZ, Vorwahl, Bland, EW, Landesregierung}
- {PLZ} ⟶ {Landesregierung}

RDM: Entwurf relationaler Schemata

Zwei alternative Methoden:

bevorzugt

- Entwickle zunächst ein ER-Diagramm, leite daraus ein relationales Schema mit Entitäten- und Beziehungstabellen ab (vgl. C. Batini, S. Ceri, S.B. Navathe. Conceptual Database Design - An Entity Relationship Approach, Benjamin/Cummings, Redwood City, Kalifornien, 1992).
- Sammle funktionale Abhängigkeiten aus der Anforderungsdefinition und erzeuge daraus ein relationales Schema in Normalform (Im Trend 1970...80).
  Ausführlich in der Literatur beschrieben (vgl. S.M. Lang, P.C. Lockemann. Datenbankeinsatz. Springer, Berlin u.a., 1995).

RDM: Relationale Algebra – Operatoren (1)

Vereinigung R ∪ S:

- Alle Tupel zweier Relationen werden in einer Ergebnisrelation zusammengefasst
- Das Ergebnis enthält keine Duplikate

$$R \cup S := \{ r \mid r \in R \vee r \in S \}$$

Relation R

ANr	*AName*	*Menge*
001	Anlasser	1.000
⋮	⋮	⋮
199	Kolben	5.000

Relation S

ANr	*AName*	*Menge*
237	Ölfilter	1.560
⋮	⋮	⋮
851	Schraube	25.000

Ergebnisrelation $R \cup S$

ANr	*AName*	*Menge*
001	Anlasser	1.000
⋮	⋮	⋮
199	Kolben	5.000
237	Ölfilter	1.560
⋮	⋮	⋮
851	Schraube	25.000

RDM: Relationale Algebra – Operatoren (2)

Differenz R \ S:

- Die Tupel zweier Relationen werden miteinander verglichen.
- Die in der ersten, nicht aber in der zweiten Relation befindlichen Tupel werden in die Ergebnisrelation aufgenommen.

$$R \setminus S := \{ r \in R \mid r \notin S \}$$

Relation R

ANr	AName	Menge
001	Anlasser	1.000
237	Ölfilter	1.560
199	Kolben	5.000

Relation S

ANr	AName	Menge
851	Schraube	25.000
232	Gummiring	2.000
001	Anlasser	1.000

Ergebnisrelation $R \setminus S$

ANr	AName	Menge
237	Ölfilter	1.560
199	Kolben	5.000

RDM: Relationale Algebra – Operatoren (3)

Durchschnitt R ∩ S:

– Alle Tupel, die sowohl in der Relationen R als auch in der Relation S enthalten sind, werden in der Ergebnisrelation zusammengefaßt.

$$R \cap S := \{ r \mid r \in R \wedge r \in S \}$$

Relation R

ANr	*AName*	*Menge*
001	Anlasser	1.000
007	Zündkerze	1.380
199	Kolben	5.000

Relation S

ANr	*AName*	*Menge*
001	Anlasser	1.000
199	Kolben	5.000
237	Ölfilter	1.560

Ergebnisrelation $R \cap S$

ANr	*AName*	*Menge*
001	Anlasser	1.000
199	Kolben	5.000

RDM: Relationale Algebra – Operatoren (4)

Kartesisches Produkt R × S:

- Alle Tupel zweier Relationen R und S werden kombinatorisch miteinander verbunden. Wenn die Relation R n Spalten und die Relation S m Spalten umfasst, dann besitzt R × S (*n*+*m*) Spalten.
- Wenn die Relation R k Zeilen und die Relation S l Zeilen umfasst, dann besitzt
  R × S (*k***l*) Zeilen.
- Um eindeutige Attributbezeichnungen in der Ergebnisrelation zu gewährleisten, müssen Attribute, die in den Relationen R und S gleich bezeichnet sind, vor der Bildung des kartesischen Produkts umbenannt werden.

$$R \times S := \{ (r_1, ..., r_n, s_1, ..., s_m) \mid (r_1, ..., r_n) \in R, (s_1, ..., s_m) \in S \}$$

- <u>Beispiel:</u>
  - Projekte × Projektdurchführung (s. nächste Folie)

RDM: Relationale Algebra – Operatoren (5)

Beispiel: Projekte × Projektdurchführung

Projekte

Nr	*Titel*	*Budget*
100	DB Fahrpläne	300.000
200	ADAC Kundenstamm	100.000
300	Telekom Statistik	200.000

Projektdurchführung (Ausschnitt)

Nr	*Kurz*
100	MFSW
200	PERS
300	MFSW

Ergebnisrelation Projekte × Projektdurchführung

Nr	*Titel*	*Budget*	*Nr2*	*Kurz*
100	DB Fahrpläne	300.000	100	MFSW
100	DB Fahrpläne	300.000	200	PERS
100	DB Fahrpläne	300.000	300	MFSW
200	ADAC Kundenstamm	100.000	100	MFSW
200	ADAC Kundenstamm	100.000	200	PERS
200	ADAC Kundenstamm	100.000	300	MFSW
300	Telekom Statistik	200.000	100	MFSW
300	Telekom Statistik	200.000	200	PERS
300	Telekom Statistik	200.000	300	MFSW

RDM: Relationale Algebra – Operatoren (6)

Join (Verbindung) R $\bowtie_{\theta}$ S:

- Eine Verbindung zwischen zwei Relationen wird in einer Kombination von kartesischem Produkt und nachfolgender Selektion (σ) gemäß des Prädikats θ hergestellt.
- Im allgemeinen Fall (*Theta-Join*) vergleicht ein (beliebiges) Prädikat θ mehrere Attribute aus den Relationen R und S (Spezialfall: Equi-Join).

$$R \bowtie_{\theta} S := \sigma_{\theta}(R \times S)$$

- <u>Beispiele:</u>
  - *Projekte* $\bowtie_{(\text{ Nr} \neq \text{Nr })}$ *Projektdurchführung* (s. nächste Folie)
  - *Projekte* $\bowtie_{(\text{ Budget} > 150000 \text{ }) \wedge (\text{ Nr} = \text{Nr })}$ *Projektdurchführung*
- Die Ergebnisrelation enthält die Zeilen des kartesischen Produkts der Relationen R und S, die σ erfüllen.

RDM: Relationale Algebra – Operatoren (7)

Beispiel: Projekte $\bowtie_{(Nr \neq Nr)}$ Projektdurchführung

Projekte

Nr	*Titel*	*Budget*
100	DB Fahrpläne	300.000
200	ADAC Kundenstamm	100.000
300	Telekom Statistik	200.000

Projektdurchführung (Ausschnitt)

Nr	*Kurz*
100	MFSW
200	PERS
300	MFSW

Ergebnisrelation

Nr	*Titel*	*Budget*	*Nr2*	*Kurz*
100	DB Fahrpläne	300.000	200	PERS
100	DB Fahrpläne	300.000	300	MFSW
200	ADAC Kundenstamm	100.000	100	MFSW
200	ADAC Kundenstamm	100.000	300	MFSW
300	Telekom Statistik	200.000	100	MFSW
300	Telekom Statistik	200.000	200	PERS

RDM: Relationale Algebra – Operatoren (8)

Join (Verbindung): Fortsetzung

- Von besonderer Bedeutung im RDM ist der *Natural Join*, da er eine Verknüpfung von Tabellen über ihre Fremdschlüsselwerte erlaubt.
  - Beispiel:
    - *Projekte* $\bowtie$ *Projektdurchführung*
      := *Projekt* $\bowtie_{Nr = Nr}$ *Projektdurchführung*
    - In diesem Fall betrachtet $\bowtie$ nur die Gleichheit zwischen Fremdschlüssel und Primärschlüssel, die den gleichen Attributnamen (*Nr*) besitzen.
- Weitere abgeleitete *Joinoperationen* (*Semi-Join, Outer-Join*, ...) und die Division zweier Relationen kommen später

RDM: Relationale Algebra – Operatoren (9)

Natural Join: Projekte $\bowtie_{(Nr = Nr)}$ Projektdurchführung

Projekte

Nr	*Titel*	*Budget*
100	DB Fahrpläne	300.000
200	ADAC Kundenstamm	100.000
300	Telekom Statistik	200.000

Projektdurchführung (Ausschnitt)

Nr	*Kurz*
100	MFSW
200	PERS
300	MFSW

Ergebnisrelation

Nr	*Titel*	*Budget*	*Nr2*	*Kurz*
100	DB Fahrpläne	300.000	100	MFSW
200	ADAC Kundenstamm	100.000	200	PERS
300	Telekom Statistik	200.000	300	MFSW

RDM: Relationale Algebra – Operatoren (10)

Projektion $\pi_{(r_{f_1}, \ldots, r_{f_n})}$ (R):

– n Spalten einer m-stelligen Relation R werden über ihren Namen ausgewählt.
– Dadurch entsteht eine n-stellige Relation *(n × m).*
– Die Reihenfolge der Spalten in der Ergebnisrelation kann definiert werden.
– Duplikatelimination in der Ergebnisrelation.

$$\pi_{(r_{f_1}, \ldots, r_{f_n})}(R) := \{ (r_{f_1}, \ldots, r_{f_n}) \mid (r_1, \ldots, r_m) \in R \}$$

– Beispiel: $\pi_{(\text{Nr, Budget})}$ (Projekte)

Projekte

Nr	*Titel*	*Budget*
100	DB Fahrpläne	300.000
200	ADAC Kundenstamm	100.000
300	Telekom Statistik	200.000

→

Ergebnisrelation $\pi_{(Nr,\, Budget)}$(Projekte)

Nr	*Budget*
100	300.000
200	100.000
300	200.000

RDM: Relationale Algebra – Operatoren (11)

Selektion σ_θ(R):

- Bestimmte Tupel einer Relation werden ausgewählt und in der Ergebnisrelation vereinigt.
- Zur Auswahl der zu übernehmenden Tupel dient das Prädikat σ : R ⟶ { `true, false` }, in dem die Attributbezeichner als Eingabevariablen dienen.
- Anwendung dieses Prädikats auf jedes Tupel der Ausgangsrelation, indem die Werte des Tupels unter den jeweiligen Attributen für die Variablen eingesetzt werden.
- In die Ergebnisrelation werden alle Tupel übernommen, für die das Prädikat den Wahrheitswert `true` liefert.

$$\sigma_\theta(R) := \{\, r \in R \mid \theta(\, r\,) \,\}$$

Relation R

ANr	*AName*
001	Anlasser
007	Zündkerze
199	Kolben

Ergebnisrelation $\sigma_{ANr < 199}(R)$

ANr	*AName*
001	Anlasser
007	Zündkerze

RDM: Relationale Algebra als Anfragesprache

Vorteil:

- Orthogonale Kombination der Konstrukte
- Einfache, mathematische Behandlung,
  z.B. $(R \ltimes S) \ltimes T = R \ltimes (S \ltimes T)$
- Einfache (naive) Implementierung möglich
- Optimierung möglich

Nachteile:

- Eingeschränkte Ausdrucksmächtigkeit
  auf Relationenebene
  (Summe, Mittelwert, Kardinalität)
- Reine Anfragesprache
- Optimierung nicht trivial