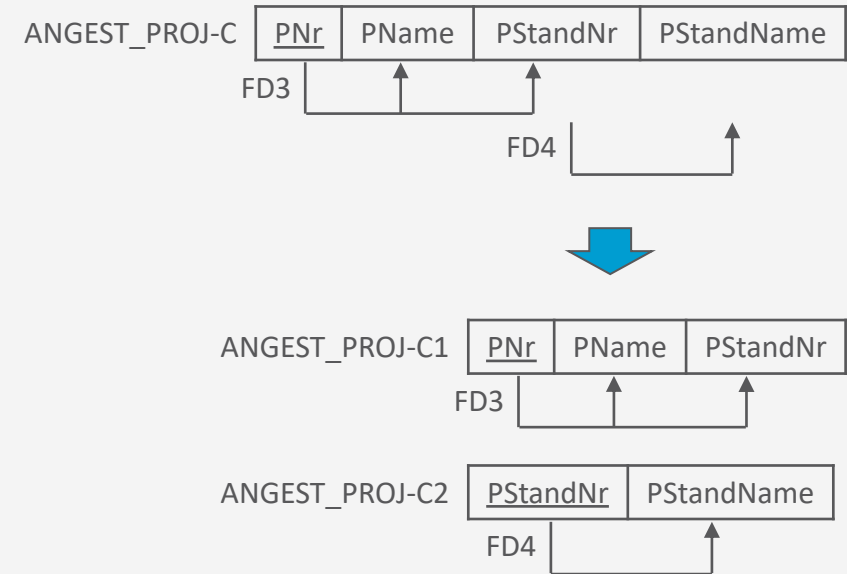


Datenbank-Entwurf

Datenbanken



Inhalte: Datenbanken (DBs)

1. Einführung

- Anwendungen
- Datenbankmanagementsysteme

2. Datenbank-Modellierung

- Entity-Relationship-Modell (ER-Modell)
- Beziehung zwischen ER und UML

3. Das relationale Modell

- Relationales Datenmodell (RM)
- Vom ER-Modell zum RM
- Relationale Algebra als Anfragesprache

4. Datenbank-Entwurf

- Funktionale Abhängigkeiten
- Normalformen

5. Structured Query Language (SQL)

- Datendefinition
- Datenmanipulation

6. Anfrageverarbeitung

- Architektur
- Indexierung
- Anfragepläne, Optimierung

7. Transaktionen

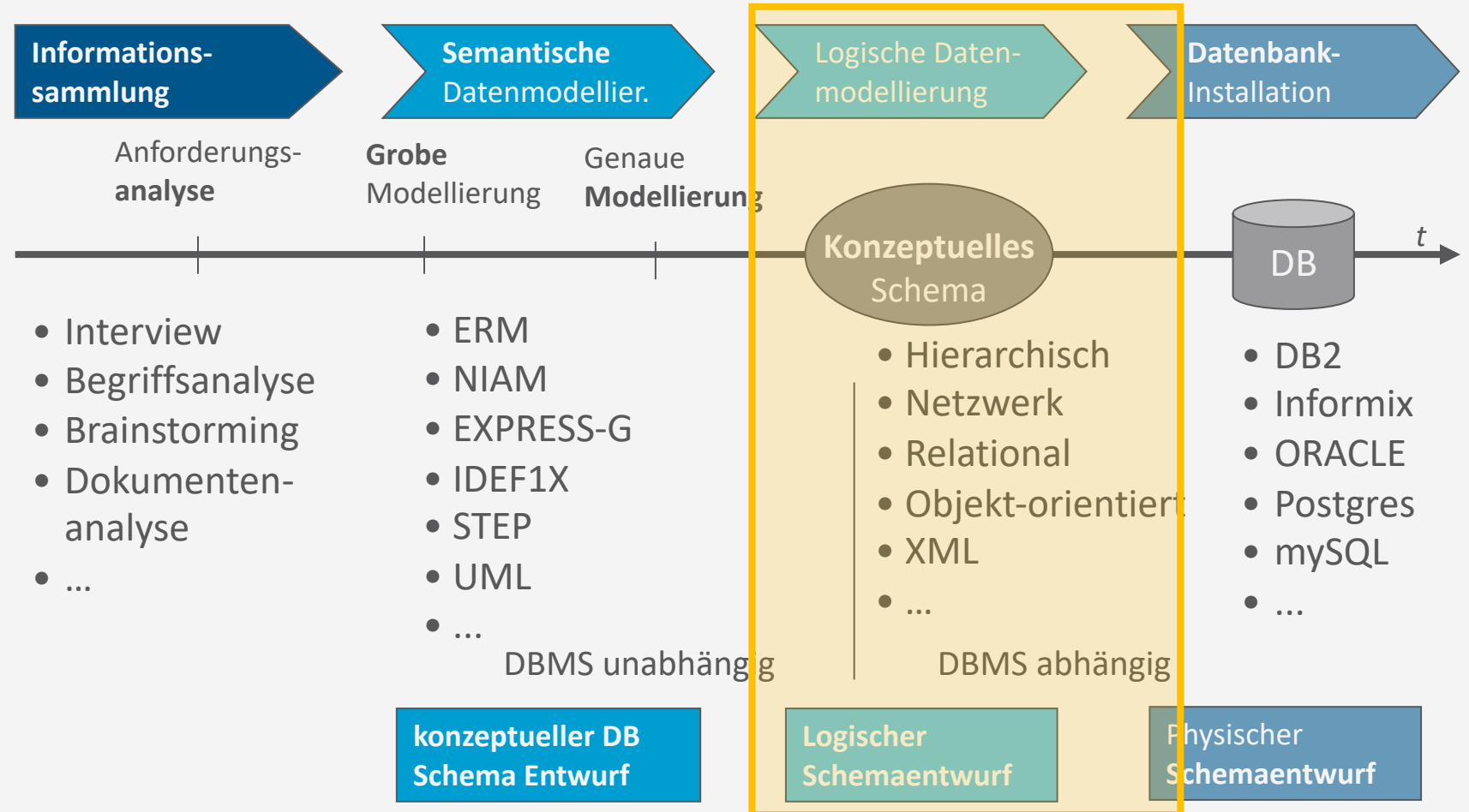
- Transaktionsverarbeitung, Schedules, Sperren
- Wiederherstellung

8. Erweiterung

- Noch offen: verteilte DBs, deduktive DBs (DataLog → Logik-Verbindung), XML, Graph-DBs

Phasen des DB-Entwurfs

- Ausblick: Von der Anwendung her
 - Teil von 2. DB-Modellierung
 - Methode: ERM
 - Teil von 3. Das relationale Datenmodell
 - Methode: relationale Modellierung
 - Teil von 4. DB-Entwurf
 - Teil von 5. SQL & Übergang zu „Hinter den Kulissen“



„Schlechte“ Relation

AName	SVNr	Geb	Adr	AbtNr	Abt	ALeitSVNr
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291, Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975, FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	Rice, Houston, TX	5	Research	333445555
Jabber, Ahmad V.	987987987	1969-03-29	Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	Stone, Houston, TX	1	Headquarters	888665555

„Schlechte“ Relation: Einfügeanomalie

AName	SVNr	Geb	Adr	AbtNr	Abt	ALeitSVNr
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291, Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975, FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	Rice, Houston, TX	5	Research	333445555
Jabber, Ahmad V.	987987987	neuer Angestellter → Information zu Abteilung (konsistent?) → Tippfehler?				87654321
Borg, James E.	888665555	1937-11-10	Stone, Houston, TX	1	Headquarters	888665555
Grawunder, M	007	1971-02-17	Barßel	1	Hädquarters	886665554
				42	Geheim	007

keine neue Abteilung ohne Mitarbeiter

„Schlechte“ Relation: Update-Anomalie

AName	SVNr	Geb	Adr	AbtNr	Abt	ALeitSVNr
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston TX	5	Research	732456732
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston TX	5	Research	732456732
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291, Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975, FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	Rice, Houston, TX	5	Research	732456732
Jabber, Ahmad V.	987987987	1969-03-29	Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	Stone, Houston, TX	1	Headquarters	888665555

Abteilung Research bekommt neuen Leiter ... hoffentlich keine Zeile vergessen

„Schlechte“ Relation: Löschanomalie

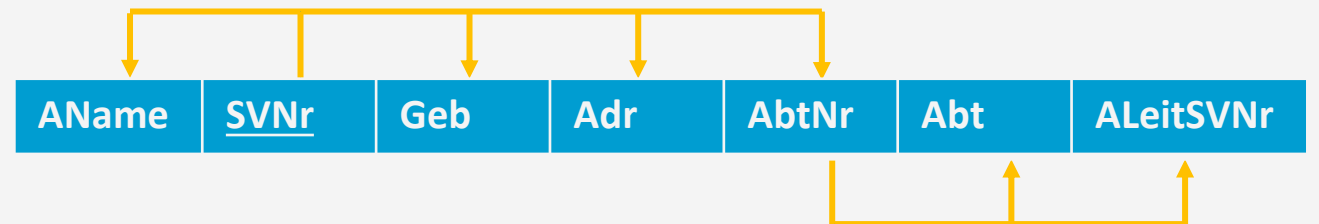
AName	SVNr	Geb	Adr	AbtNr	Abt	ALeitSVNr
Smith, John B.	123456789	1965-01-09	731 Fondren, Houston TX	5	Research	333445555
Wong, Franklin T.	333445555	1955-12-08	638 Voss, Houston TX	5	Research	333445555
Zelaya, Alicia J.	999887777	1968-07-19	3321 Castle, Spring, TX	4	Administration	987654321
Wallace, Jennifer S.	987654321	1941-06-20	291, Berry, Bellaire, TX	4	Administration	987654321
Narayan, Ramesh K.	666884444	1962-09-15	975, FireOak, Humble, TX	5	Research	333445555
English, Joyce A.	453453453	1972-07-31	Rice, Houston, TX	5	Research	333445555
Jabber, Ahmad V.	987987987	1969-03-29	Dallas, Houston, TX	4	Administration	987654321
Borg, James E.	888665555	1937-11-10	Stone, Houston, TX	1	Headquarters	888665555

James E. Borg geht in den Ruhestand. Es gibt aber noch keinen Nachfolger ... Wo ist die Abteilung Headquarters hin??

„Schlechte“ Relation

- Problem: Kombination aus Angestellten- mit Abteilungsinformation
- Einfüge-Anomalie
 - Neuer Angestellter → Information zu Abteilung konsistent?
 - Keine neue Abteilung ohne Mitarbeiter
- Update-Anomalie
 - Änderung des Abteilungsleiters → Update in allen Angestellten-Einträgen?
- Lösch-Anomalie
 - Letzter Angestellter einer Abteilung gelöscht → Abteilung verschwindet

- Was fällt auf?
 - Es tauchen Redundanzen in den Tupeln auf
 - Verschiedene **Abhängigkeiten** in der Relation
 - Änderungen an den Daten können zu Anomalien führen, da
 - Informationen an mehreren Stellen verändert werden müssen
 - Abhängigkeiten nicht berücksichtigt werden



Ziele der relationalen Entwurfstheorie

- Bewertung der Qualität eines Relationenschemas
 - Redundanz
 - Viel Redundanz erhöht Gefahr für Anomalien, vor allem bei Updates
 - Einhaltung von Konsistenzbedingungen
 - **Funktionale Abhängigkeiten**, anhand derer Konsistenz geprüft werden kann
- **Normalformen** als Gütekriterium
- Gegebenenfalls Verbesserung eines Relationenschemas durch
 - Synthesealgorithmus
 - Zerlegungsalgorithmus
- Meist gute Qualität bei aus validen ER-Diagrammen erstellte DB-Schemata

Überblick: 4. Datenbankentwurf

A. Funktionale Abhängigkeiten

- Definition, Schlüssel, Ableitungen
- Attributhülle, kanonische Überdeckung

B. Normalformen

- Zerlegung von Relationen
- Exkurs: NF^2 ; 1NF, 2NF, 3NF, BCNF, 4NF, 5NF
- Synthesealgorithmus, Zerlegungsalgorithmus

C. Datenqualität

- Datenqualitätsprobleme, Dimensionen der Datenqualität

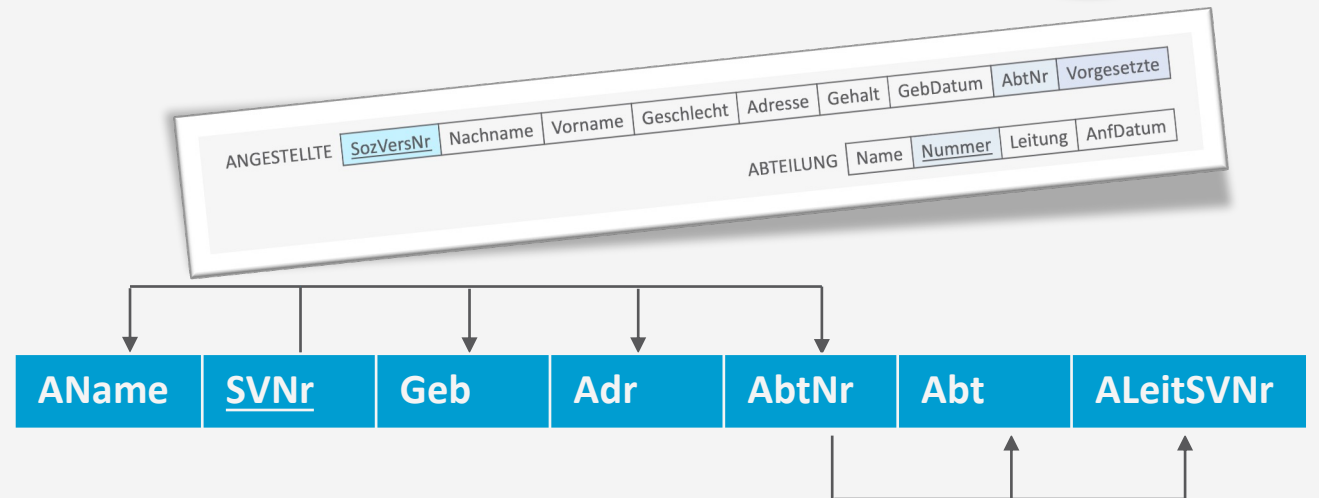
Funktionale Abhängigkeiten

- $R(A_1, \dots, A_n)$ sei ein Schema, α und β seien Attributteilmenge von R
- FD: $\alpha \rightarrow \beta$ sei eine funktionale Abhängigkeit (*functional dependency*), genau dann wenn

$$\forall t_1, t_2 \in r(R) : \text{Wenn } t_1[\alpha] = t_2[\alpha] \text{ gilt, gilt auch } t_1[\beta] = t_2[\beta]$$
- D.h., Werte von α bestimmen eindeutig Werte von β
- FD heißt *trivial*, wenn $\beta \subseteq \alpha$
- Für ANGEST_ABTEILUNG gilt:
 - F1: $\{SVNr\} \rightarrow \{AName, Geb, Adr, AbtNr\}$
 - F2: $\{AbtNr\} \rightarrow \{Abt, ALeitSVNr\}$
 - Relationen abgeleitet aus ER-Modell
- Wenn α ein Schlüssel ist, dann gilt $\alpha \rightarrow \beta$ für alle möglichen β aus R

Folgt aus $\alpha \rightarrow \beta$
auch $\beta \rightarrow \alpha$?

Was ist, wenn α
ein Schlüssel ist?



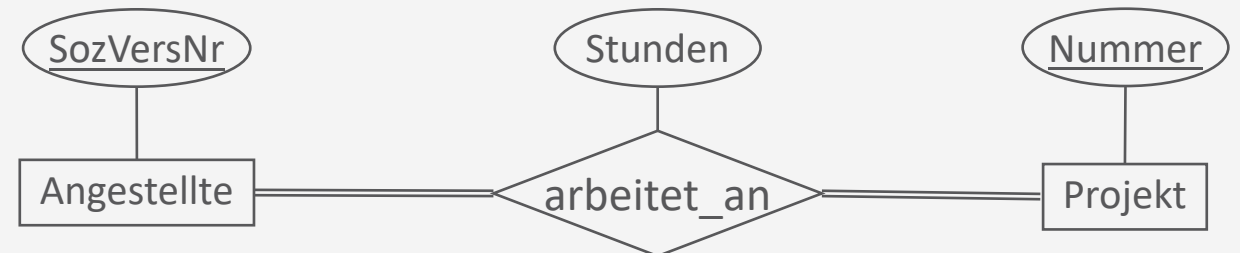
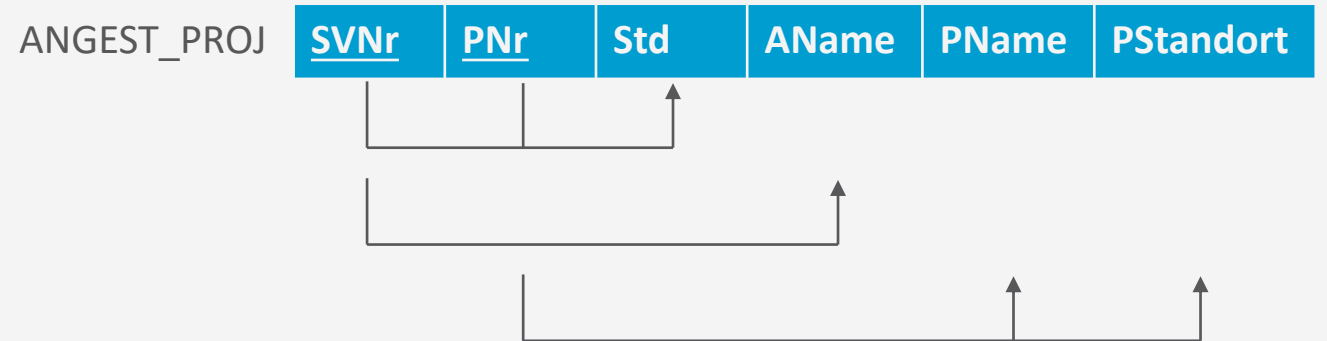
Funktionale Abhängigkeiten

- Für ANGEST_PROJ gilt
 - F3: $\{SVNr, PNr\} \rightarrow \{Std\}$
 - F4: $\{SVNr\} \rightarrow \{AName\}$
 - F5: $\{PNr\} \rightarrow \{PName, PStandort\}$
- Relationen abgeleitet aus ER-Modell:

ANGESTELLTE	<u>SozVersNr</u>	Nachname	Vorname	Geschlecht	Adresse	Gehalt	GebDatum	AbtNr	Vorgesetzte
-------------	------------------	----------	---------	------------	---------	--------	----------	-------	-------------

ARBEITET_AN	<u>ProjNr</u>	<u>SozVersNr</u>	Stunden
-------------	---------------	------------------	---------

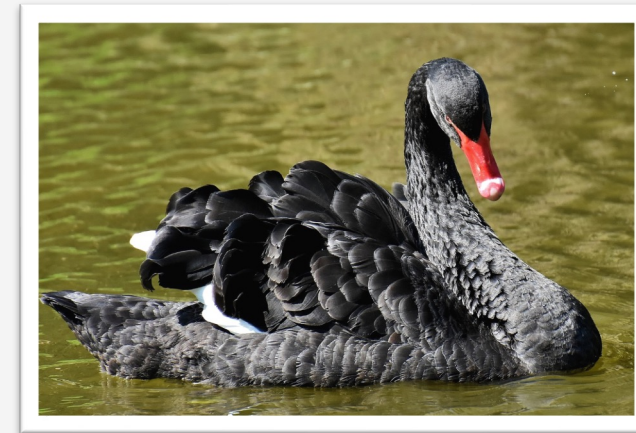
PROJEKT	<u>Nummer</u>	Name	Standort	AbtNr
---------	---------------	------	----------	-------



Wo kommen die FDs her?

- Können die nicht einfach aus den Daten abgeleitet werden?
- Beispiel: Tierart → Farbe?

Datum	Tierart	Farbe
12.3.2010	Schwan	weiß
14.3.2010	Fuchs	rot
17.3.2010	Schwan	weiß



"Not all swans
are white."

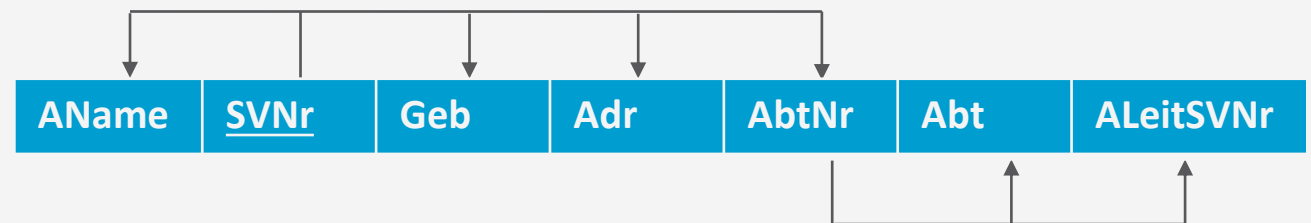
- FDs können nicht aus den Daten (Extension) abgeleitet werden
- FDs sind Eigenschaften des **Relationenschemas** (Intension)
- Sie müssen vom **DB-Designer** definiert werden

Schlüssel

- $\alpha \subseteq R$ ist ein **Superschlüssel**, falls gilt:
 - $\alpha \rightarrow R$
- β ist **voll funktional abhängig** von α genau dann, wenn gilt
 - $\alpha \rightarrow \beta$ und
 - α kann nicht mehr verkleinert werden (α minimal), d.h.
 - $\forall A \in \alpha$ folgt, dass $(\alpha - \{A\}) \rightarrow \beta$ nicht gilt, oder kürzer
 - $\forall A \in \alpha : \neg((\alpha - \{A\}) \rightarrow \beta)$
 - Notation für volle funktionale Abhängigkeit: $\alpha \rightarrow \cdot \beta$
- $\alpha \subseteq R$ ist ein **Kandidatenschlüssel**, falls gilt:
 - $\alpha \rightarrow \cdot R$

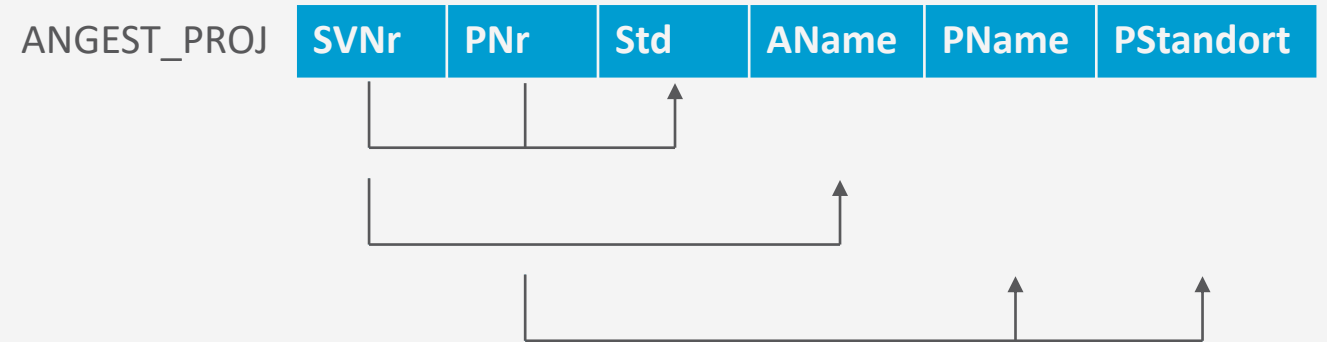
Schlüssel: Beispiel 1

- Beispiel: Für ANGEST_ABT gelten $F = \{F1, F2\}$
 - $F1: \{SVNr\} \rightarrow \{AName, Geb, Adr, AbtNr\}$
 - $F2: \{AbtNr\} \rightarrow \{Abt, ALeitSVNr\}$
- $\{SVNr, AbtNr\} \rightarrow \{SVNr, AbtNr, AName, Geb, Adr, Abt, ALeitSVNr\}$
 - $\{SVNr, AbtNr\}$ ein Superschlüssel?
 - Ja, da alle Attribute eindeutig durch $\{SVNr, AbtNr\}$ bestimmt werden
 - $\{SVNr, AbtNr, AName, Geb, Adr, Abt, ALeitSVNr\}$ voll funktional abhängig von $\{SVNr, AbtNr\}$?
 - Nein, da $\{SVNr, AbtNr\}$ verkleinert werden kann: $\{SVNr\}$ reicht aus
- $\{SVNr, AbtNr\}$ Kandidatenschlüssel?
 - Nein, da volle funktionale Abhängigkeit nicht gilt



Schlüssel: Beispiel 2

- Für ANGEST_PROJ gilt
 - F3: $\{SVNr, PNr\} \rightarrow \{Std\}$
 - F4: $\{SVNr\} \rightarrow \{AName\}$
 - F5: $\{PNr\} \rightarrow \{PName, PStandort\}$
 - Volle funktionale Abhängigkeiten
 - $\{SVNr\} \rightarrow \cdot \{SVNr, AName\}$
 - Kein Superschlüssel, da z.B. *Std* nicht allein von *SVNr* abhängig
 - $\{PNr\} \rightarrow \cdot \{PNr, PName, PStandort\}$
 - Kein Superschlüssel, da z.B. *Std* nicht allein von *PNr* abhängig
 - $\{SVNr, PNr\} \rightarrow \cdot \{SVNr, PNr, Std, AName, PName, PStandort\}$
 - Super- und Kandidatenschlüssel, da $\{SVNr, PNr\} \rightarrow \cdot ANGEST_PROJ$



Wie sehen weitere
Superschlüssel aus?

Wie sieht es mit deren
voller funktionaler
Abhängigkeit aus?

Herleitung funktionaler Abhängigkeiten

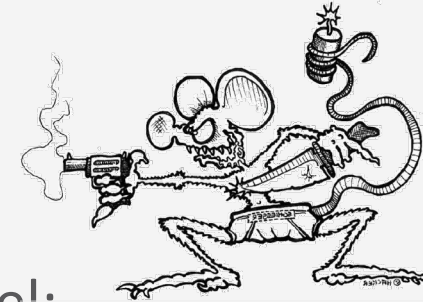
- Aus gegebenen FDs können weitere FDs abgeleitet werden
 - Mittels Inferenzregeln
- Transitive Hülle F^+ :
 - Auch manchmal Abschluss (engl. *closure*) genannt
 - Gegeben eine Menge von FDs F
 - F^+ : Menge aller FDs, die mit Inferenzregeln abgeleitet werden können
- Es gibt sechs Inferenzregeln (RATZAP)
 - $IR1$: Reflexivitätsregel
 - $IR2$: Augmentationsregel
 - $IR3$: Transitivitätsregel
 - $IR4$: Zerlegungsregel
 - $IR5$: Additive oder Vereinigungsregel
 - $IR6$: Pseudotransitive Regel

$IR1-3$ auch unter **Armstrong-Axiomen** bekannt

- Vollständig: Erzeugen nur gültige FDs
 - Korrekt: Jede mögliche FD lässt sich herleiten
- $IR4-6$ lassen sich aus $IR1-3$ ableiten
- Erleichtern Ableitungen

Die sechs Inferenzregeln (RATZAP)

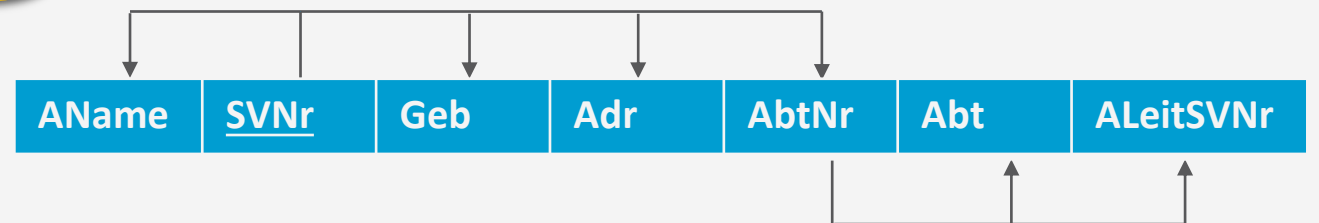
- Reflexivitätsregel:
 - *IR1*: Falls $Y \subseteq X$, dann $X \rightarrow Y$
 - Eine Attributmengem bestimmt sich immer selbst oder eine ihrer Teilmengen
- Augmentationsregel:
 - *IR2*: Falls $X \rightarrow Y$, dann $XZ \rightarrow YZ$
 - Hinzufügen von Attributen auf beiden Seiten führt zu weiterer Regel
- Transitivitätsregel:
 - *IR3*: Falls $\{X \rightarrow Y, Y \rightarrow Z\}$, dann $X \rightarrow Z$
- Zerlegungsregel:
 - *IR4*: Falls $X \rightarrow YZ$, dann $X \rightarrow Y$ und $X \rightarrow Z$
 - Attribute auf der rechten Seite können entfernt und FDs in Teilmengen zerlegt werden
- Additive oder Vereinigungsregel:
 - *IR5*: Falls $\{X \rightarrow Y, X \rightarrow Z\}$, dann $X \rightarrow YZ$
 - Gegenstück zu **Z** (*IR4*):
Regel wieder zusammenfassen
- Pseudotransitive Regel:
 - *IR6*: Falls $\{X \rightarrow Y, WY \rightarrow Z\}$, dann $WX \rightarrow Z$
 - Transitivität im Kontext



Beispiel für die Anwendung von Inferenzregeln

- Beispiel: $F = \{F1, F2\}$
 - F1: $\{SVNr\} \rightarrow \{AName, Geb, Adr, AbtNr\}$
 - F2: $\{AbtNr\} \rightarrow \{Abt, ALeitSVNr\}$
 - Zusätzlich können u.a. abgeleitet werden
 - F3: $\{SVNr\} \rightarrow \{Abt, ALeitSVNr\}$
 - IR3
 - F4: $\{SVNr\} \rightarrow \{Adr, AbtNr\}$
 - IR4
 - F5: $\{SVNr\} \rightarrow \{SVNr\}$
 - IR1

Was für FDs können abgeleitet werden?



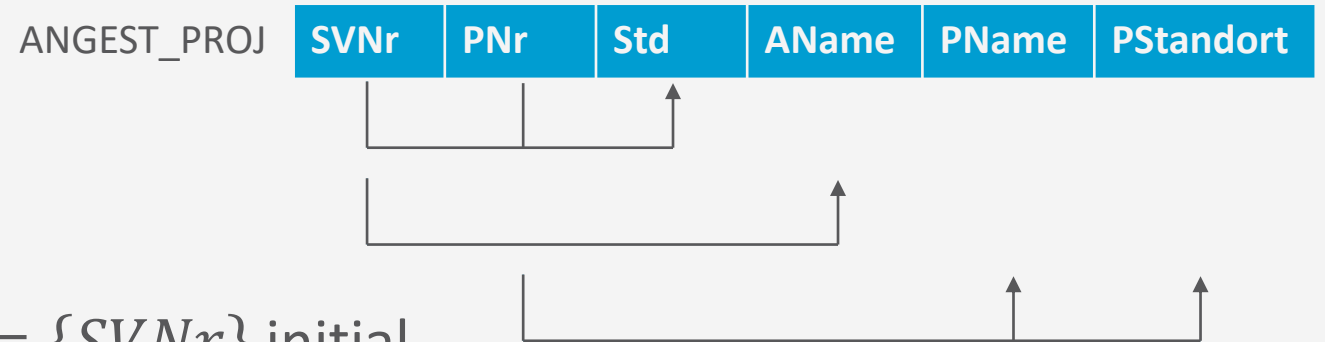
Attributhülle

- **Attributhülle** α^+ für eine Menge von Attributen α
 - Alle Attribute, die von α funktional abhängen
- Berechnung
 - Eingabe: Menge von FDs F , Menge von Attributen α
 - Ausgabe: Vollständige Menge von Attributen α^+ , für die gilt: $\alpha \rightarrow \alpha^+$
 - Vorgehen:
 - Initialisierung: $\alpha^+ \leftarrow \alpha$
 - Für jede FD $\beta \rightarrow \gamma$ in F
 - Wenn β in α^+ vorkommt, dann füge γ zu α^+ hinzu
 - Gebe α^+ aus

```
ATTRHÜLLE( $F, \alpha$ )  
   $\alpha^+ \leftarrow \alpha$   
  while  $\alpha^+$  geändert do  
    for each  $\beta \rightarrow \gamma \in F$  do  
      if  $\beta \subseteq \alpha^+$  then  
         $\alpha^+ \leftarrow \alpha^+ \cup \gamma$   
  return  $\alpha^+$ 
```

Attributhülle: Beispiel

- Für ANGEST_PROJ gilt
 - F3: $\{SVNr, PNr\} \rightarrow \{Std\}$
 - F4: $\{SVNr\} \rightarrow \{AName\}$
 - F5: $\{PNr\} \rightarrow \{PName, PStandort\}$
- Attributhülle $\alpha = \{SVNr\}$ mit $\alpha^+ = \alpha = \{SVNr\}$ initial
 - Betrachte F3, F4, F5:
 - Betrachte F3 mit $\beta = \{SVNr, PNr\}$: $\beta \not\subseteq \alpha^+$ (keine Änderung)
 - Betrachte F4 mit $\beta = \{SVNr\}$: $\beta \subseteq \alpha^+$, damit $\alpha^+ = \{SVNr, AName\}$
 - Betrachte F5 mit $\beta = \{PNr\}$: $\beta \not\subseteq \alpha^+$ (keine Änderung)
 - Änderung in α^+ , also nochmal: Betrachte F3, F4, F5
 - Keine Änderung mehr in α^+
 - Ausgabe: $\alpha^+ = \{SVNr, AName\}$

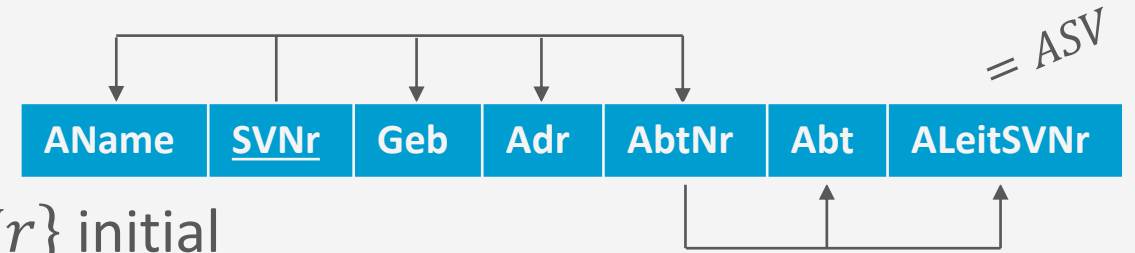


```

ATTRHÜLLE( $F, \alpha$ )
 $\alpha^+ \leftarrow \alpha$ 
while  $\alpha^+$  geändert do
  for each  $\beta \rightarrow \gamma \in F$  do
    if  $\beta \subseteq \alpha^+$  then
       $\alpha^+ \leftarrow \alpha^+ \cup \gamma$ 
return  $\alpha^+$ 
  
```

Attributhülle: Beispiel

- $F1: \{SVNr\} \rightarrow \{AName, Geb, Adr, AbtNr\}$
- $F2: \{AbtNr\} \rightarrow \{Abt, ASV\}$
- Attributhülle $\alpha = \{SVNr\}$, mit $\alpha^+ = \alpha = \{SVNr\}$ initial
 - Betrachte F2, F1
 - Betrachte F2 mit $\beta = \{AbtNr\}$: $\beta \not\subseteq \alpha^+$ (keine Änderung)
 - Betrachte F1 mit $\beta = \{SVNr\}$: $\beta \subseteq \alpha^+$, damit $\alpha^+ = \alpha^+ \cup \{AName, Geb, Adr, AbtNr\}$
 - Änderung in α^+ ; also nochmal: Betrachte F2, F1
 - Betrachte F2 mit $\beta = \{AbtNr\}$: $\beta \subseteq \alpha^+$, damit $\alpha^+ = \alpha^+ \cup \{Abt, ASV\}$
 - Betrachte F1 mit $\beta = \{SVNr\}$: $\beta \subseteq \alpha^+$, aber keine Änderung in α^+
 - Änderung in α^+ , also nochmal: Betrachte F2, F1
 - Keine Änderung mehr in α^+
 - Ausgabe: $\alpha^+ = \{SVNr, AName, Geb, Adr, AbtNr, Abt, ASV\}$



```

ATTRHÜLLE( $F, \alpha$ )
 $\alpha^+ \leftarrow \alpha$ 
while  $\alpha^+$  geändert do
    for each  $\beta \rightarrow \gamma \in F$  do
        if  $\beta \subseteq \alpha^+$  then
             $\alpha^+ \leftarrow \alpha^+ \cup \gamma$ 
return  $\alpha^+$ 
    
```

Attributhülle

- **Attributhülle** α^+ für eine Menge von Attributen α
 - Alle Attribute, die von α funktional abhängen
- Nutzen
 - Berechnung der transitiven Hülle F^+ :
 - Für alle $\gamma \subseteq R$ und alle $S \subseteq \gamma^+$ enthält F^+ die FD $\gamma \rightarrow S$
 - Bestimmung von Superschlüsseln:
 - α Superschlüssel, wenn α^+ alle Attribute von R enthält
 - Überprüfung (voll) funktionaler Abhängigkeiten:
 - $\alpha \rightarrow \beta$ gilt, wenn $\beta \subseteq \alpha^+$

```
ATTRHÜLLE( $F, \alpha$ )  
   $\alpha^+ \leftarrow \alpha$   
  while  $\alpha^+$  geändert do  
    for each  $\beta \rightarrow \gamma \in F$  do  
      if  $\beta \subseteq \alpha^+$  then  
         $\alpha^+ \leftarrow \alpha^+ \cup \gamma$   
  return  $\alpha^+$ 
```

Redundanz in FDs & Kanonische Überdeckung

- Menge von FDs F kann redundante FDs enthalten oder FDs mit überflüssigen Attributen
 - Macht Überprüfung von Konsistenzen bei Änderungen im DB-Zustand aufwendiger als nötig
 - Gesucht: Minimale Menge an FDs, die ausreichend sind, um F^+ zu beschreiben
- **Kanonische Überdeckung F_C** , welche folgende drei Kriterien erfüllt
1. $F_C \equiv F$, d.h. $F_C^+ = F^+$
 2. In F_C existieren keine FDs $\alpha \rightarrow \beta$, die überflüssige Attribute in α oder β enthalten
 - Kein überflüssiges Attribut in α : $\forall A \in \alpha : (F_C - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - \{A\}) \rightarrow \beta\} \neq F_C$
 - Test: Wenn $\beta \subseteq (\alpha - \{A\})^+$, dann A überflüssig (kann aus α entfernt werden)
 - Kein überflüssiges Attribut in β : $\forall B \in \beta : (F_C - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - \{B\})\} \neq F_C$
 - Test: Wenn $B \in \alpha^+$, dann B überflüssig (kann aus β entfernt werden)
 3. Jede linke Seite einer FD in F_C ist einzigartig
 - Additive Regel sukzessive anwenden: Wenn FDs $\alpha \rightarrow \beta, \alpha \rightarrow \gamma$, ersetze durch $\alpha \rightarrow \beta\gamma$

Kanonische Überdeckung: Berechnung

- Eingabe: Menge von FDs F
 - Ausgabe: Kanonische Überdeckung F_C (Menge von FDs, welche die drei Kriterien erfüllt)
 - Initialisiere F_C mit F
1. Führe für jede FD $\alpha \rightarrow \beta \in F_C$ die **Linksreduktion** durch:
 - Überprüfe für alle $A \in \alpha$:
 - Wenn $\beta \subseteq (\alpha - \{A\})^+$ bzgl. F_C , ersetze $\alpha \rightarrow \beta$ mit $(\alpha - \{A\}) \rightarrow \beta$ in F_C
 2. Führe für jede FD $\alpha \rightarrow \beta \in F_C$ die **Rechtsreduktion** durch:
 - Überprüfe für alle $B \in \beta$:
 - Wenn $B \in \alpha^+$ bzgl. $(F_C - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - \{B\})\}$, ersetze $\alpha \rightarrow \beta$ mit $\alpha \rightarrow (\beta - \{B\})$ in F_C
 3. Entferne die FDs der Form $\alpha \rightarrow \emptyset$ aus F_C
 4. Ersetze alle FDs der Form $\alpha \rightarrow \beta_1, \dots, \alpha \rightarrow \beta_n$ durch $\alpha \rightarrow (\beta_1 \cup \dots \cup \beta_n)$ in F_C

Kanonische Überdeckung: Pseudocode

- Eingabe:
Menge von FDs F
- Ausgabe:
Kanonische Überdeckung F_C
- Initialisiere F_C mit F
 1. Linksreduktion
 2. Rechtsreduktion
 3. FDs mit leerem β
entfernen
 4. FDs mit gleichem α
zusammenfassen

```

KANON( $F$ )
   $F_C \leftarrow F$ 
  for each  $\alpha \rightarrow \beta \in F_C$  do
    for each  $A \in \alpha$  do
      if  $\beta \subseteq \text{ATTRHÜLLE}(F_C, \alpha - \{A\})$  then
         $F_C \leftarrow (F_C - \{\alpha \rightarrow \beta\}) \cup \{(\alpha - \{A\}) \rightarrow \beta\}$ 
  for each  $\alpha \rightarrow \beta \in F_C$  do
    for each  $B \in \beta$  do
      if  $B \in \text{ATTRHÜLLE}((F_C - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - \{B\})\}, \alpha)$  then
         $F_C \leftarrow (F_C - \{\alpha \rightarrow \beta\}) \cup \{\alpha \rightarrow (\beta - \{B\})\}$ 
  for each  $\alpha \rightarrow \beta \in F_C$  do
    if  $\beta = \emptyset$  then
       $F_C \leftarrow F_C - \{\alpha \rightarrow \beta\}$ 
  while  $\exists \alpha \rightarrow \beta_1, \dots, \alpha \rightarrow \beta_n$  do
     $F_C \leftarrow (F_C - \{\alpha \rightarrow \beta_1, \dots, \alpha \rightarrow \beta_n\}) \cup \{\alpha \rightarrow (\beta_1 \cup \dots \cup \beta_n)\}$ 
  return  $F_C$ 

```

Kanonische Überdeckung

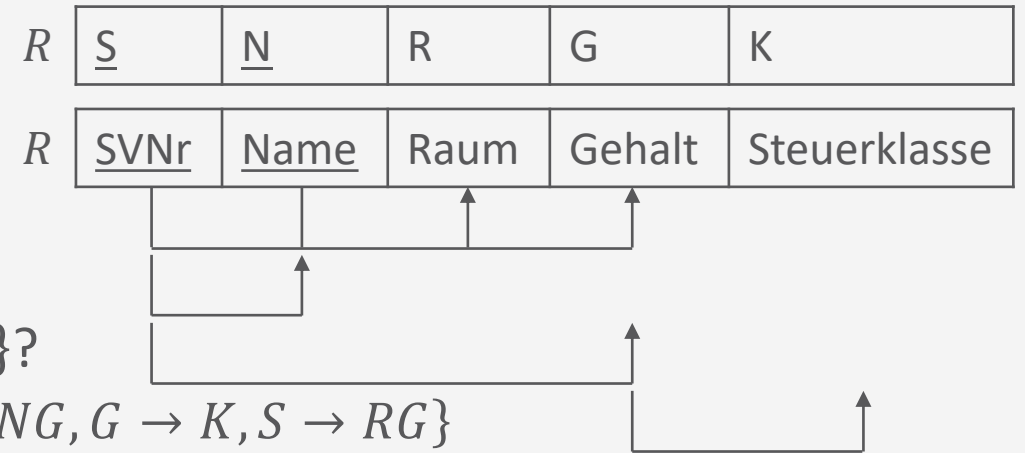
- $F = \{S \rightarrow NG, G \rightarrow K, SN \rightarrow RG\} = F_C$

1. Linksreduktion: $SN \rightarrow RG$ in F_C testen

- S überflüssig, d.h., $\{R, G\} \subseteq \{N\}^+ = \{N\}$?
 - Nein, daher keine Änderung in F_C
- N überflüssig, d.h., $\{R, G\} \subseteq \{S\}^+ = \{S\} \cup \{N, G, K, R\}$?
 - Ja, daher $F_C \leftarrow (F_C - \{SN \rightarrow RG\}) \cup \{S \rightarrow RG\} = \{S \rightarrow NG, G \rightarrow K, S \rightarrow RG\}$

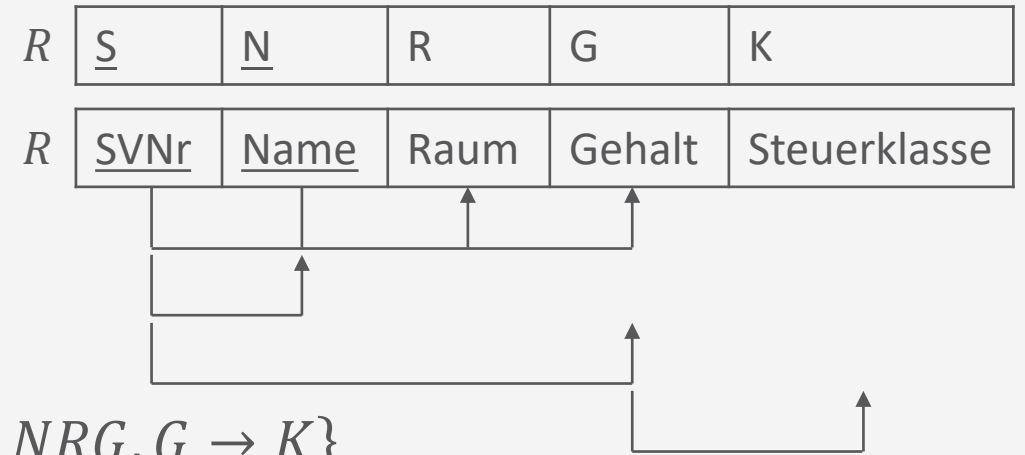
2. Rechtsreduktion: $S \rightarrow NG, S \rightarrow RG$ in F_C mit reduzierter FD testen

- $S \rightarrow NG$: N überflüssig in $\{S \rightarrow G, G \rightarrow K, S \rightarrow RG\}$, d.h., $\{N\} \subseteq \{S\}^+ = \{S\} \cup \{G, K, R\}$?
 - Nein, daher keine Änderung in F_C
- $S \rightarrow NG$: G überflüssig in $\{S \rightarrow N, G \rightarrow K, S \rightarrow RG\}$, d.h., $\{G\} \subseteq \{S\}^+ = \{S\} \cup \{N, R, G\} \cup \{K\}$?
 - Ja, daher $F_C \leftarrow (F_C - \{S \rightarrow NG\}) \cup \{S \rightarrow N\} = \{S \rightarrow N, G \rightarrow K, S \rightarrow RG\}$
- $S \rightarrow RG$: beide nicht überflüssig



Kanonische Überdeckung

- $F = \{S \rightarrow NG, G \rightarrow K, SN \rightarrow RG\} = F_C$
- $F_C = \{S \rightarrow N, G \rightarrow K, S \rightarrow RG\}$ nach Schritt 1 + 2
- 3. FDs mit leerem β entfernen
 - Keine Änderung
- 4. FDs mit gleichem α zusammenfassen
 - $F_C \leftarrow (F_C - \{S \rightarrow N, S \rightarrow RG\}) \cup \{S \rightarrow NRG\} = \{S \rightarrow NRG, G \rightarrow K\}$
 - Ergebnis: $F_C = \{S \rightarrow NRG, G \rightarrow K\}$



Ist F_C eindeutig?

Kanonische Überdeckung: Abstraktes Beispiel

- Relation über A, B, C
 - $F = \{A \rightarrow BC, B \rightarrow CA, C \rightarrow A\} = F_C$
- Schritt mit Effekt: Rechtsreduktion
 - $A \rightarrow BC$ zuerst betrachten
 - $A \rightarrow BC: B \in \{A\}^+ = \{A\} \cup \{C\}$?
 - Nein, keine Änderung
 - $A \rightarrow BC: C \in \{A\}^+ = \{A\} \cup \{C\}$?
 - Ja, daher $F_C = \{A \rightarrow B, B \rightarrow CA, C \rightarrow A\}$
 - $B \rightarrow CA: C \in \{B\}^+ = \{B\} \cup \{A\}$?
 - Nein, keine Änderung
 - $B \rightarrow CA: A \in \{B\}^+ = \{B\} \cup \{A\}$?
 - Ja, daher $F_C = \{A \rightarrow B, B \rightarrow C, C \rightarrow A\}$
- Relation über A, B, C
 - $F = \{A \rightarrow BC, B \rightarrow CA, C \rightarrow A\} = F_C$
- Schritt mit Effekt: Rechtsreduktion
 - $B \rightarrow CA$ zuerst betrachten
 - $B \rightarrow CA: C \in \{B\}^+ = \{B\} \cup \{A, C\}$?
 - Ja, daher $F_C = \{A \rightarrow BC, B \rightarrow A, C \rightarrow A\}$
 - $A \rightarrow BC: B \in \{A\}^+ = \{A\} \cup \{C\}$?
 - Nein, keine Änderung
 - $A \rightarrow BC: C \in \{A\}^+ = \{A\} \cup \{B\}$?
 - Nein, keine Änderung
 - Ergebnis: $F_C = \{A \rightarrow BC, B \rightarrow A, C \rightarrow A\}$

Kanonische Überdeckung nicht immer eindeutig

Zwischenzusammenfassung

- Probleme bei schlecht gebauten Relationen:
 - Einfüge-Anomalien, Update-Anomalien, Lösch-Anomalien
- Bewertung der Qualität über Konsistenzbedingungen durch FDs und Redundanz
- FDs
 - Erlauben Schlüssel zu bestimmen: Relation dann voll funktional abhängig
 - Transitive Hülle: Herleitung über Inferenzregeln (RATZAP)
 - Attributhülle: Menge von Attributen, die funktional abhängig sind von gegebenen Attributen
 - Erlaubt Berechnung von Superschlüsseln und transitiver Hülle, Test von funktionalen Abhängigkeiten
 - Kanonische Überdeckung zur Minimierung der Menge der FDs durch Entfernung von überflüssigen Attributen und FDs
 - Vorgehen: Rechtsreduktion, Linksreduktion, leere FDs löschen, gleiche α Regeln zusammenfassen
 - Nutzt auch die Attributhülle in ihren Berechnungen

Überblick: 4. Datenbankentwurf

A. Funktionale Abhängigkeiten

- Definition, Schlüssel, Ableitungen
- Attributhülle, kanonische Überdeckung

B. Normalformen

- Zerlegung von Relationen
- (Exkurs: NF^2), 1NF, 2NF, 3NF, BCNF, 4NF, 5NF
- Synthesealgorithmus, Zerlegungsalgorithmus

C. Datenqualität

- Datenqualitätsprobleme, Dimensionen der Datenqualität

Erinnerung: Ziele der relationalen Entwurfstheorie

- Bewertung der Qualität eines Relationenschemas
 - Redundanz
 - Viel Redundanz erhöht Gefahr für Anomalien, vor allem bei Updates
 - Einhaltung von Konsistenzbedingungen
 - **Funktionale Abhängigkeiten**, anhand derer Konsistenz geprüft werden kann
- **Normalformen** als Gütekriterium
- Gegebenenfalls Verbesserung eines Relationenschemas durch
 - Synthesealgorithmus
 - Zerlegungsalgorithmus
- Meist gute Qualität bei aus validen ER-Diagrammen erstellten DB-Schemata

Normalisierung von Datenbank-Schemata & Normalformen (NF)

- Normalisierung bietet:
 - Normalformbedingungen, um NF zu testen
 - Vorgehen, um NF zu erreichen
 - Grundsätzliches Vorgehen:
 - **Prüfung** eines Relationenschemas auf eine Normalform
 - Wenn nicht erfüllt: **Zerlegung** in neue Relationenschemata, bis gewünschte Normalform erreicht ist
 - Je höher die NF, desto weniger Redundanz

Zerlegung von Schemata

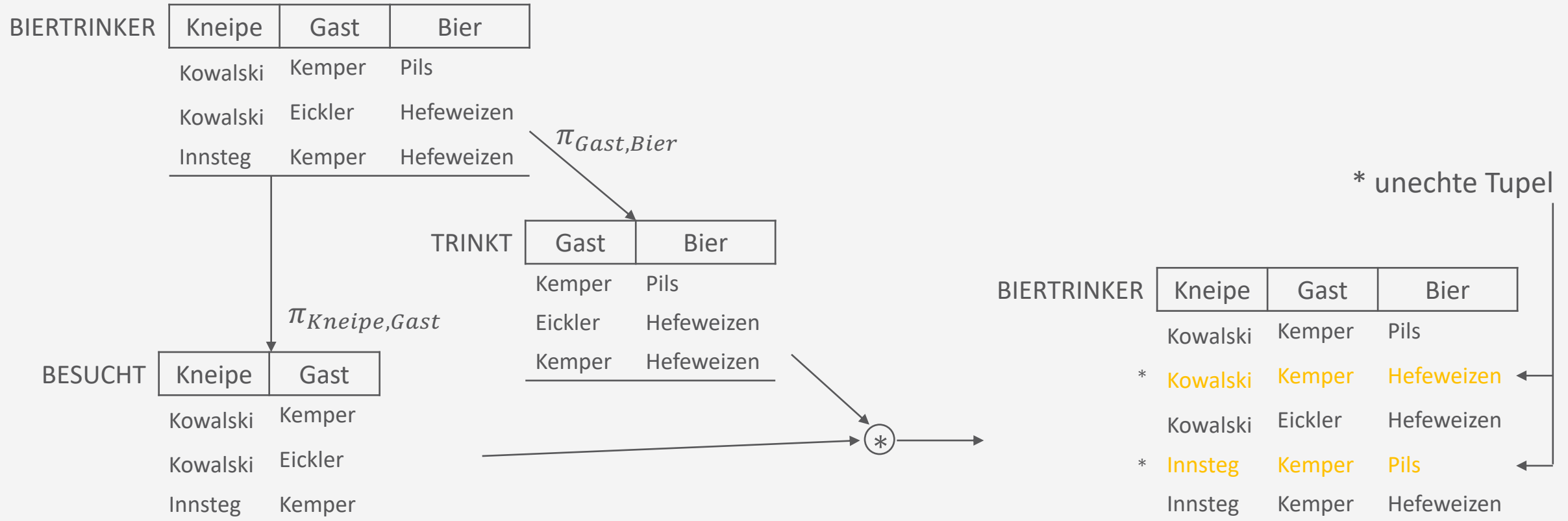
- Zur Herstellung einer NF:
Zerlegung eines Schemas R in Menge von Schemata $D = \{R_1, \dots, R_n\}$
 - Attributerhaltung der Zerlegung D :
 - Jedes Attribut aus R erscheint in mindestens einem R_i
 - Vereinigung der Attributmengen aller R_i entspricht damit der Attributmenge von R
 - Korrektheitskriterien (beide sollen möglichst gelten)
 1. **Verlustlosigkeit** bzw. Eigenschaft des nicht-additiven JOINS
 - Die im ursprünglichen Relationenzustand $r(R)$ enthaltenen Informationen müssen aus den Zuständen der neuen Schemata D rekonstruierbar sein \rightarrow keine unechten Tupel entstehen lassen
 2. **Abhängigkeitserhaltung**
 - Die für R geltenden Abhängigkeiten müssen auf die Schemata in D übertragbar sein
- ❖ Zusätzlich soll gelten, dass möglichst wenig **Redundanz** in den Daten herrscht

Verlustlosigkeit bzw. nicht-additiver JOIN

- Zerlegung $D = \{R_1, \dots, R_m\}$ von R ist verlustlos in Bezug auf die Abhängigkeitsmenge F in R , wenn für jede Relation $r(R)$, die F erfüllt, gilt
$$* \left(\pi_{R_1}(r), \dots, \pi_{R_m}(r) \right) = r$$
 - * bezeichnet hier den NATURAL JOIN
 - Hinreichende Bedingung für $D = \{R_1, R_2\}$ von R
 - Wenn $(R_1 \cap R_2) \rightarrow R_1 \in F^+$ oder $(R_1 \cap R_2) \rightarrow R_2 \in F^+$, dann ist D verlustlos
- D.h. JOINS über die zerlegten Relationen erzeugen keine „unechten“ Tupel
 - Informationsgehalt von R und D bleibt gleich

Beispiel

- Zerlegung, die **nicht** die Eigenschaft des nicht-additiven JOINs erfüllt

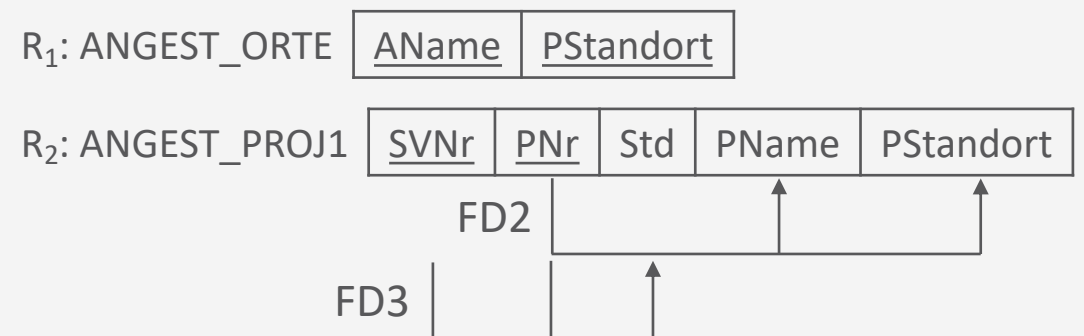
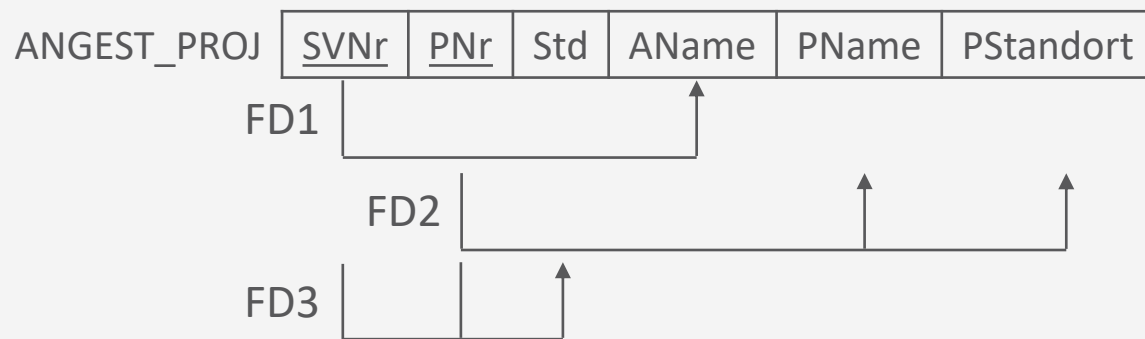


Nicht-additiver JOIN-Test

- Input: Relationenschema R , Zerlegung $D = \{R_1, R_2, \dots, R_m\}$ von R , Menge von FDs F
 1. Erzeuge Matrix S : Zeile i für jede Relation R_i in D , Spalte j für jedes Attribut A_j in R
 2. Setze $S(i, j) := b_{ij}$ für alle Matrixeinträge
 3. Für jede Zeile i , jede Spalte j : Wenn Attribut A_j zu Relation R_i gehört, dann setze $S(i, j) := a_j$
 4. Wiederhole die folgende Schleife, bis keine Änderung mehr in S :
 - Für jede FD $X \rightarrow Y$ in F , für alle Zeilen in S mit den gleichen Symbolen in den Spalten, die Attributen in X entsprechen: Setze die Symbole in jeder Spalte, die einem Attribut in Y entsprechen, in allen diesen Zeilen wie folgt gleich: Falls eine der Zeilen ein Symbol "a" in der Spalte enthält,
 - Dann setze die Werte der anderen Zeilen in dieser Spalte auf dieses Symbol "a"
 - Sonst: Wähle eines der Symbole "b", die in einer Zeile für das Attribut erscheinen, und setze die Werte in den anderen Zeilen in der Spalte auf dieses Symbol "b";
 5. Besteht eine Zeile vollständig aus Symbolen "a", dann weist die Zerlegung die Eigenschaft des nicht-additiven JOINS auf, im anderen Fall hat die Zerlegung diese Eigenschaft nicht

Nicht-additiver JOIN-Test: Beispiel 1

- Gegeben $R = \text{ANGEST_PROJ} = \{SVNr, AName, PNr, PName, PStandort, Std\}$
 - $F = \{\{SVNr\} \rightarrow \{AName\}, \{PNr\} \rightarrow \{PName, PStandort\}, \{SVNr, PNr\} \rightarrow \{Std\}\}$
- Zerlegung $D = \{R_1, R_2\}$
 - $R_1 = \text{ANGEST_ORTE} = \{AName, PStandort\}$
 - $R_2 = \text{ANGEST_PROJ1} = \{SVNr, PNr, Std, PName, PStandort\}$

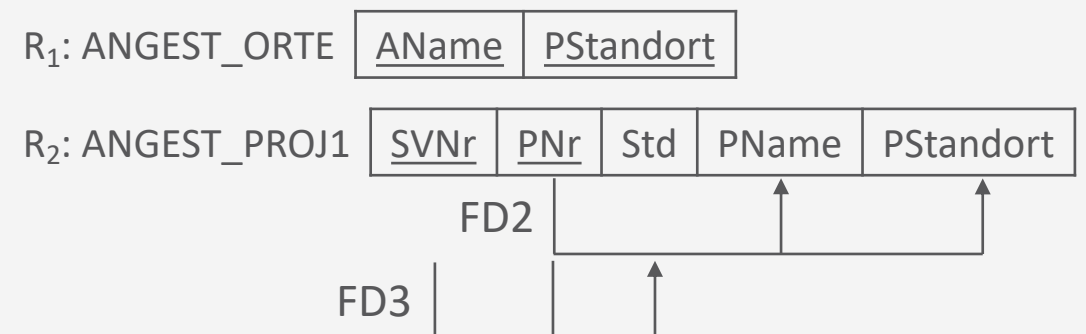


Nicht-additiver JOIN-Test: Beispiel 1

- Matrix:

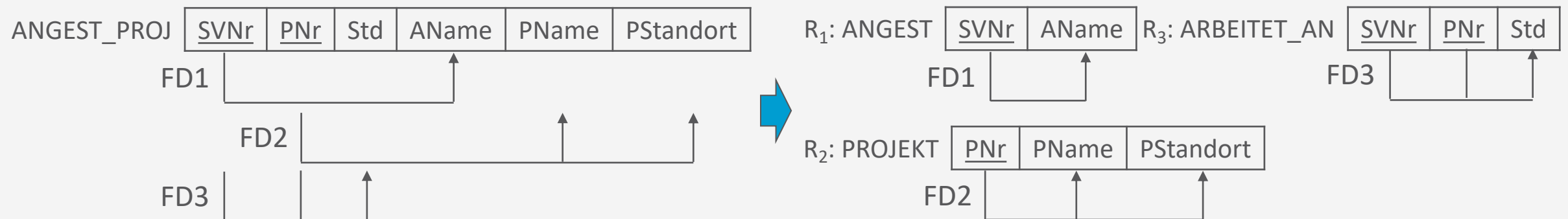
	SVNr	AName	PNr	PName	PStandort	Std
R_1	b_{11}	a_2	b_{13}	b_{14}	a_5	b_{16}
R_2	a_1	b_{22}	a_3	a_4	a_5	a_6

- Keine Änderung über die Schleife
- Keine Zeile nur mit „a“, d.h. keine Zerlegung mit gewünschter Eigenschaft
→ Additiver JOIN!



Nicht-additiver JOIN-Test: Beispiel 2

- Alternative Zerlegung von *ANGEST_PROJ*
 - Gegeben $R = ANGEST_PROJ = \{SVNr, AName, PNr, PName, PStandort, Std\}$
 - $F = \{\{SVNr\} \rightarrow \{AName\}, \{PNr\} \rightarrow \{PName, PStandort\}, \{SVNr, PNr\} \rightarrow \{Std\}\}$
 - Zerlegung $D = \{R_1, R_2, R_3\}$
 - $R_1 = ANGEST = \{SVNr, AName\}$
 - $R_2 = PROJEKT = \{PNr, PName, PStandort\}$
 - $R_3 = ARBEITET_AN = \{SVNr, PNr, Std\}$



Nicht-additiver JOIN – Test: Beispiel 2

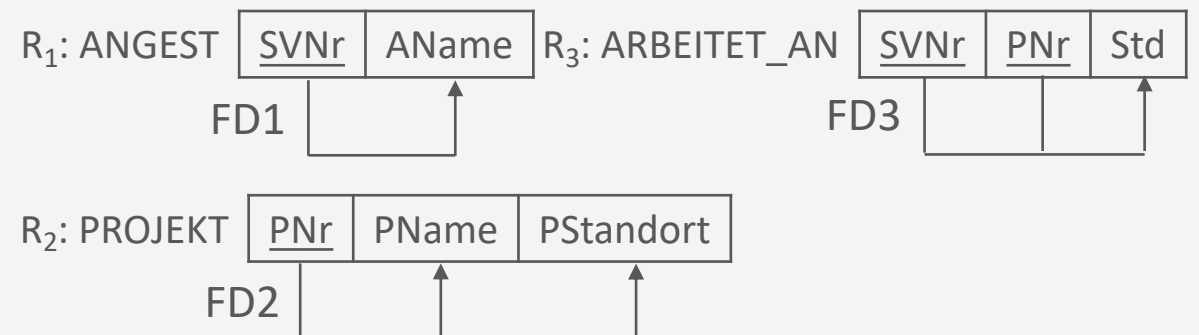
- Matrix
 - Nach Schritt 3:

	SVNr	AName	PNr	PName	PStandort	Std
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	b_{32}	a_3	b_{34}	b_{35}	a_6

- Nach **FD1** und **FD2** in Schritt 4

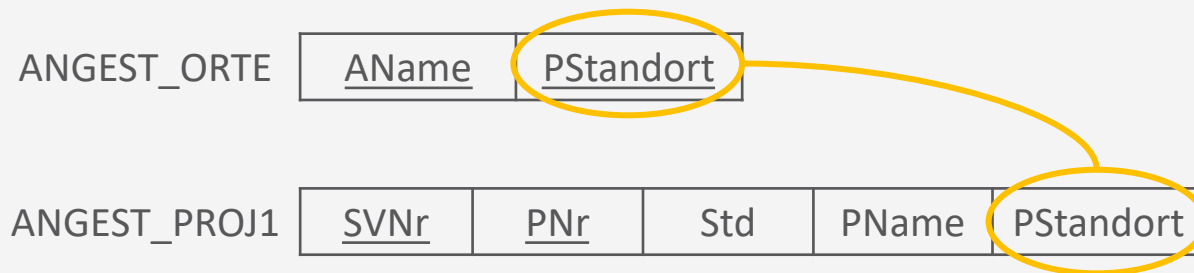
	SVNr	AName	PNr	PName	PStandort	Std
R_1	a_1	a_2	b_{13}	b_{14}	b_{15}	b_{16}
R_2	b_{21}	b_{22}	a_3	a_4	a_5	b_{26}
R_3	a_1	a_2	a_3	a_4	a_5	a_6

Zeile R_3 nur mit „a“ Symbolen
→ nicht-additive JOIN-Zerlegung



Nebenbemerkung: Unechte Tupel

- Auch aus Equijoins über Nicht-Schlüssel-Attribute können **unechte (*spurious*) Tupel** entstehen
 - Abhilfe:
 - Vermeidung gleichlautender Attribute, die keine Schlüssel sind
 - Wenn dies unvermeidbar ist: kein Join darüber bzw. Join nur über Schlüsselattribute
 - Beispiel:
 - *ANGEST_ORTE* * *ANGEST_PROJ1* (Natural Join)



Beispiel

ANGEST_PROJ1

<u>SVNr</u>	<u>PNr</u>	Std	PName	<u>PStandort</u>
123456789	1	32.5	Product X	Bellaire
123456789	2	7.5	Product Y	Sugarland
666884444	3	40.0	Product Z	Houston
453453453	1	20.0	Product X	Bellaire
453453453	2	20.0	Product Y	Sugarland

ANGEST_ORTE

<u>AName</u>	<u>PStandort</u>
Smith, John B.	Bellaire
Smith, John B.	Sugarland
Narayan, Ramesh K.	Houston
English, Joyce A.	Bellaire
English, Joyce A.	Sugarland
Wong, Franklin T.	Sugarland
Wong, Franklin T.	Houston

AName	SVNr
Smith, John B.	123456789
Wong, Franklin T.	333445555
Wallace, Jennifer S.	987654321
Narayan, Ramesh K.	666884444
English, Joyce A.	453453453
Borg, James E.	888665555

Stafford
Stafford
Stafford
Stafford
Houston
Houston

	SVNr	PNr	Std	PName	PStandort	AName
	123456789	1	32.5	Product X	Bellaire	Smith, John B.
*	123456789	1	32.5	Product X	Bellaire	English, Joyce A.
	123456789	2	7.5	Product Y	Sugarland	Smith, John B.
*	123456789	2	7.5	Product Y	Sugarland	English, Joyce A.
*	123456789	2	7.5	Product Y	Sugarland	Wong, Franklin T.
	666884444	3	40.0	Product Z	Houston	Narayan, Ramesh K.
*	666884444	3	40.0	Product Z	Houston	Wong, Franklin T.
*	666884444	3	40.0	Product Z	Houston	Wallace, Jennifer S.
*	666884444	3	40.0	Product Z	Houston	Borg, James E.
*	453453453	1	20.0	Product X	Bellaire	Smith, John B.
	453453453	1	20.0	Product X	Bellaire	English, Joyce A.
						⋮

* unechte Tupel

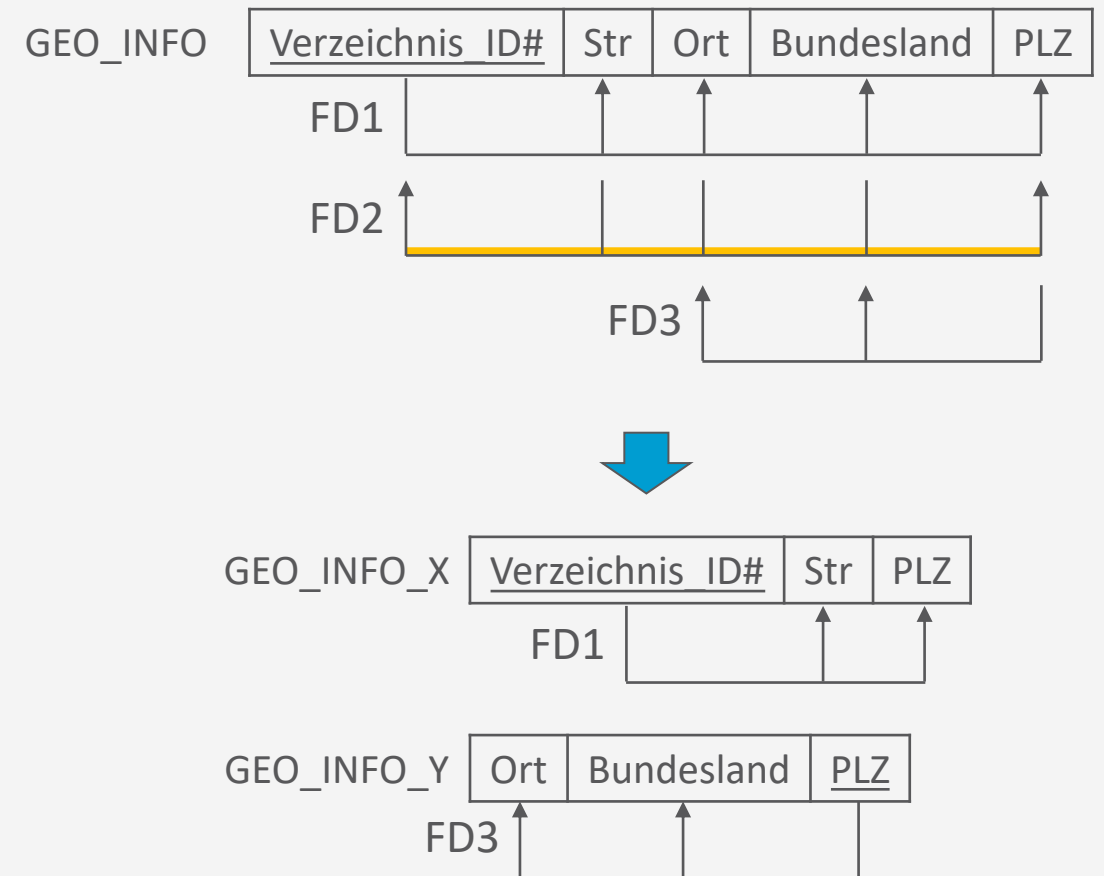
Abhängigkeitswahrung

- Jede in F spezifizierte FD $X \rightarrow Y$
 - Soll direkt in einem der R_i aus der Zerlegung D erscheinen oder
 - (Indirekt) aus den Abhängigkeiten, die in den R_i gelten, abgeleitet werden können
 - Warum ist das wichtig?
 - Wenn Abhängigkeiten verloren gehen, dann ist das Einfügen inkonsistenter Tupel möglich
- Formale Betrachtung
 - Gegeben sei eine Menge von FDs F in R
 - Dann ist die Projektion $\pi_{R_i}(F)$ von F auf R_i (für alle R_i aus D) die Menge von Abhängigkeiten $X \rightarrow Y$ in F^+ , für die alle Attribute $X \cup Y$ in R_i vorkommen
 - Die Zerlegung $D = \{R_1, \dots, R_m\}$ von R heißt in Bezug auf F **abhängigkeitswährend**, wenn die Vereinigung der Projektionen von F auf jedes R_i in D mit F äquivalent ist, i.e.,

$$\left(\pi_{R_1}(F) \cup \dots \cup \pi_{R_m}(F) \right)^+ = F^+$$

Abhängigkeitswahrung: Beispiel 1

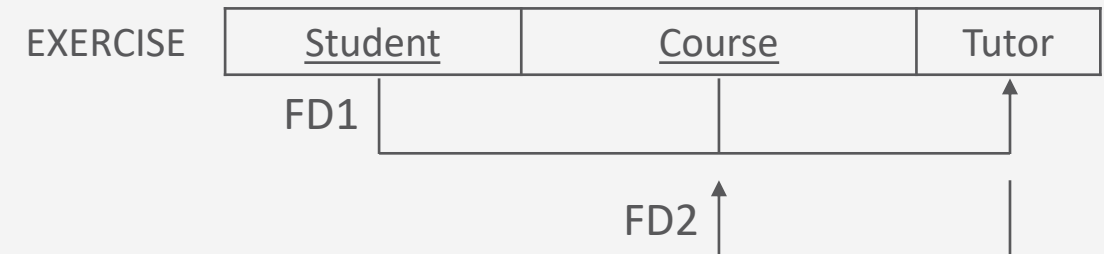
- Zerlegung, die nicht alle FDs erhält ...
 - Bei der unten gezeigten Zerlegung von GeoInfo in die Relationenschemata GeoInfoX und GeoInfoY geht FD2 verloren
 - (Verzeichnis_ID# könnte z.B. für GPS-Koordinaten stehen)
 - (FD3 stimmt nur unter der Annahme, dass PLZ nicht länder- bzw. ortsübergreifend sind)



Abhängigkeitswahrung: Beispiel 2

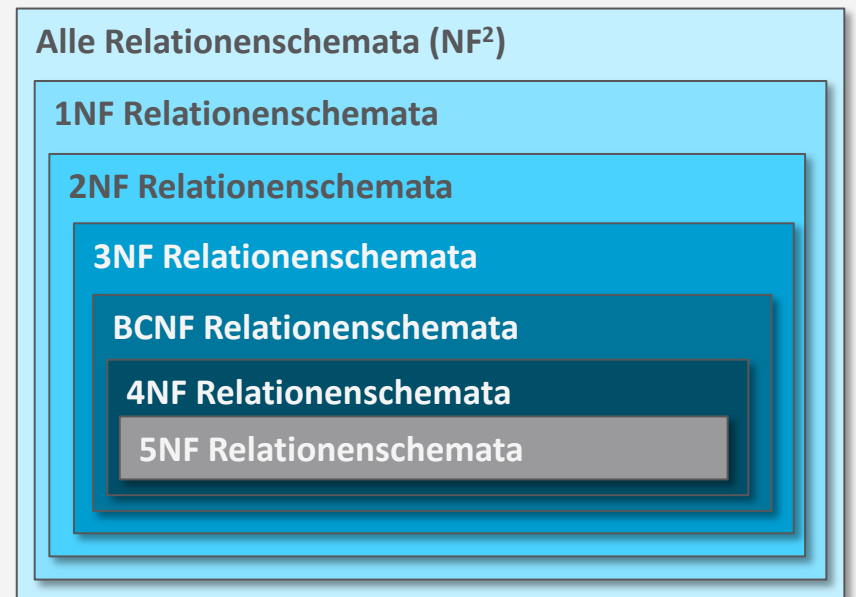
- Gegeben ist die Relation *EXERCISE* mit den FDs
 - $FD1 : \{Student, Course\} \rightarrow \{Tutor\}$
 - $FD2 : \{Tutor\} \rightarrow \{Course\}$
- Mögliche Zerlegungen
 - $(Student, Tutor)$ und $(Student, Course)$
 - $(Course, Tutor)$ und $(Student, Course)$
 - $(Course, Tutor)$ und $(Student, Tutor)$
 - FD1 geht natürlich bei allen möglichen Zerlegungen verloren, denn man bräuchte für sie alle drei Attribute in genau einem Relationenschema

EXERCISE	<u>Student</u>	<u>Course</u>	Tutor
	Narayan	Information Systems	Mark
	Smith	Information Systems	Navathe
	Smith	Operating Systems	Ammar
	Smith	Formal Languages	Schulz
	Wallace	Information Systems	Mark
	Wallace	Operating Systems	Ahamad
	Wong	Information Systems	Otte
	Zelaya	Information Systems	Navathe



Normalformen

Zerlegungen von Relationen

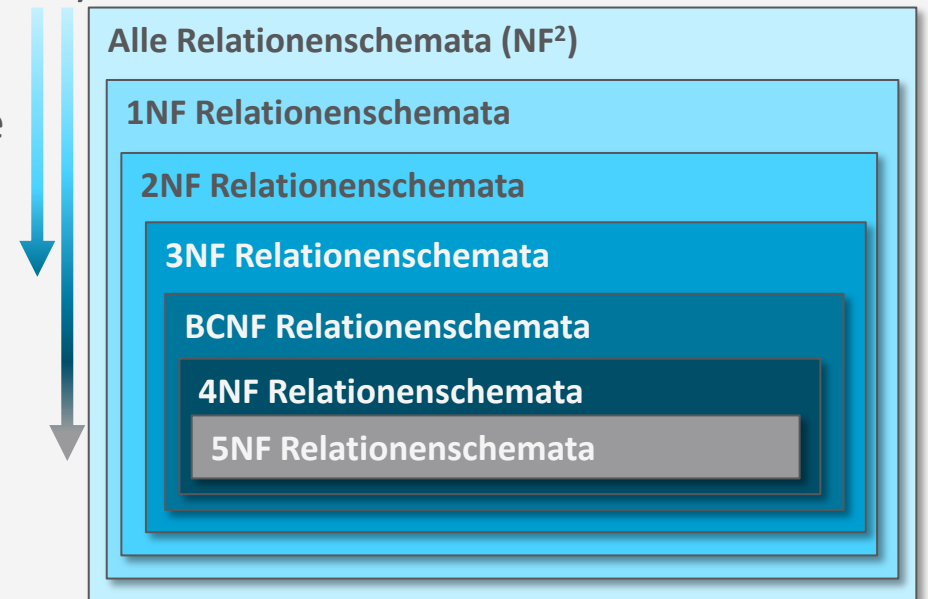


Historie

- Normalisierungsverfahren: 1972 von Codd vorgeschlagen
 - Unterzieht DB-Schemata einer Reihe von Tests, ob sie einer bestimmten NF genügen
 - Ursprünglich 1. – 3. Normalform (1NF, 2NF, 3NF)
 - Dann Verschärfung der 3. NF: Boyce Codd Normalform (BCNF)
 - Später: 4. und 5. Normalform
- NFs enthalten einander
 - Siehe Abbildung rechts

Abhängigkeitserhaltende
Zerlegung
(bis 3NF)

Verlustlose Zerlegung
(bis 5NF)



Nebenbemerkung: Non-first Normal Form (NF²)

- Relationen mit
 - Nicht-atomaren Einträgen
 - Listen
 - Zusammengesetzte Einträge
 - Geschachtelten Relationen

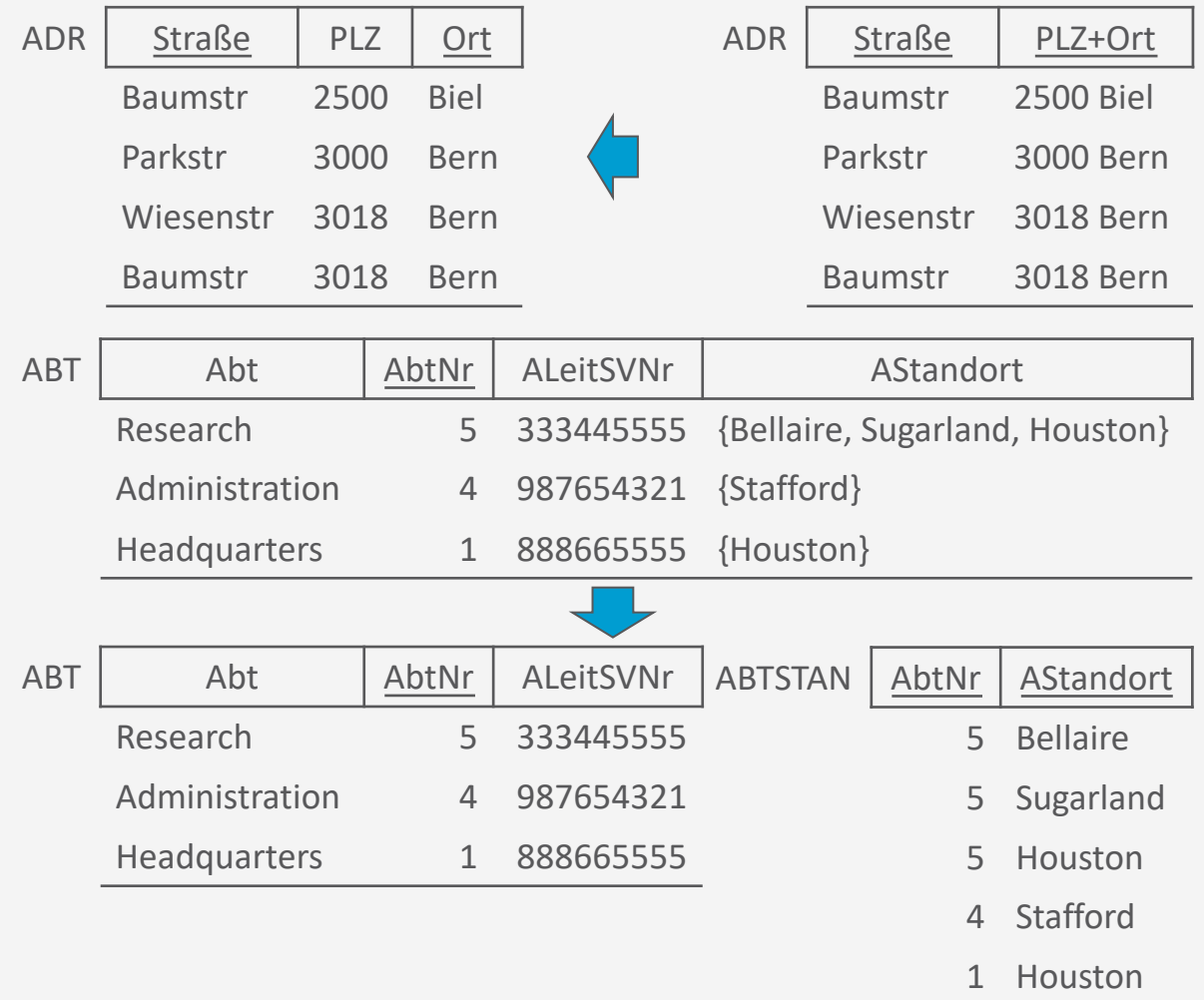
ELTERN	Vater	Mutter	Kinder	
			KName	KAlter
	Johann	Martha	Else	5
			Lucie	3
	Johann	Maria	Theo	3
			Josef	1
	Heinz	Martha	Cleo	9

ADR	<u>Straße</u>	<u>PLZ+Ort</u>
	Baumstr	2500 Biel
	Parkstr	3000 Bern
	Wiesenstr	3018 Bern
	Baumstr	3018 Bern

ABT	Abt	<u>AbtNr</u>	ALeitSVNr	AStandort
	Research	5	333445555	{Bellaire, Sugarland, Houston}
	Administration	4	987654321	{Stafford}
	Headquarters	1	888665555	{Houston}

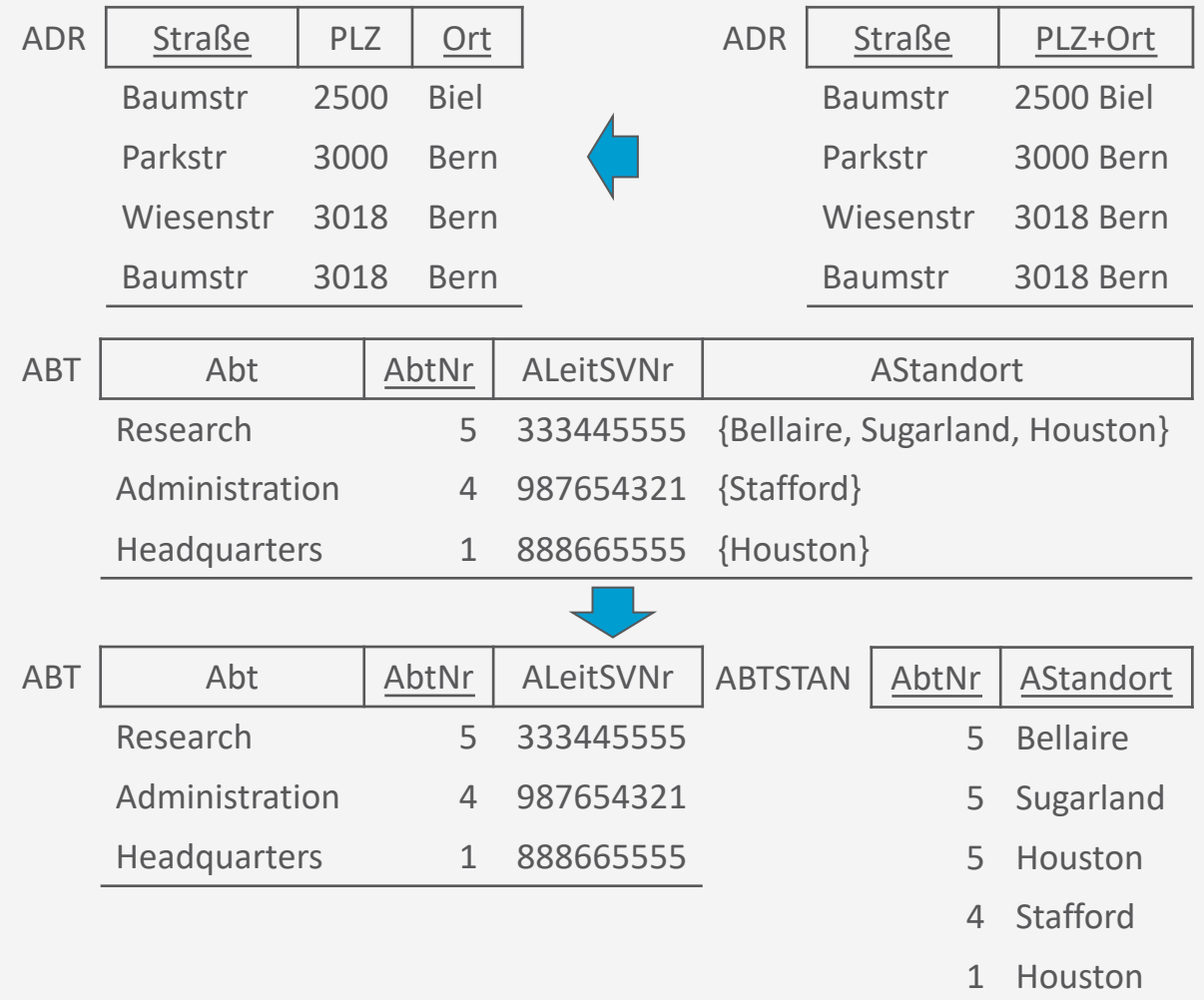
1. Normalform

- Relationenschema R ist in 1. Normalform (1NF), wenn
 - Domänen der Attribute von R nur **atomare Werte** enthalten
- Erzeugung:
 - Erstelle für jedes nicht-atomare Attribut oder für verschachtelte Schemata ein neues Relationenschema
 - Bei Übersetzung aus einem ER-Diagramm
 - *Zusammengesetzte* Attribute in seinen Einzelteilen als Attribute übernehmen
 - *Mehrwertige* Attribute in eigene Relation auslagern



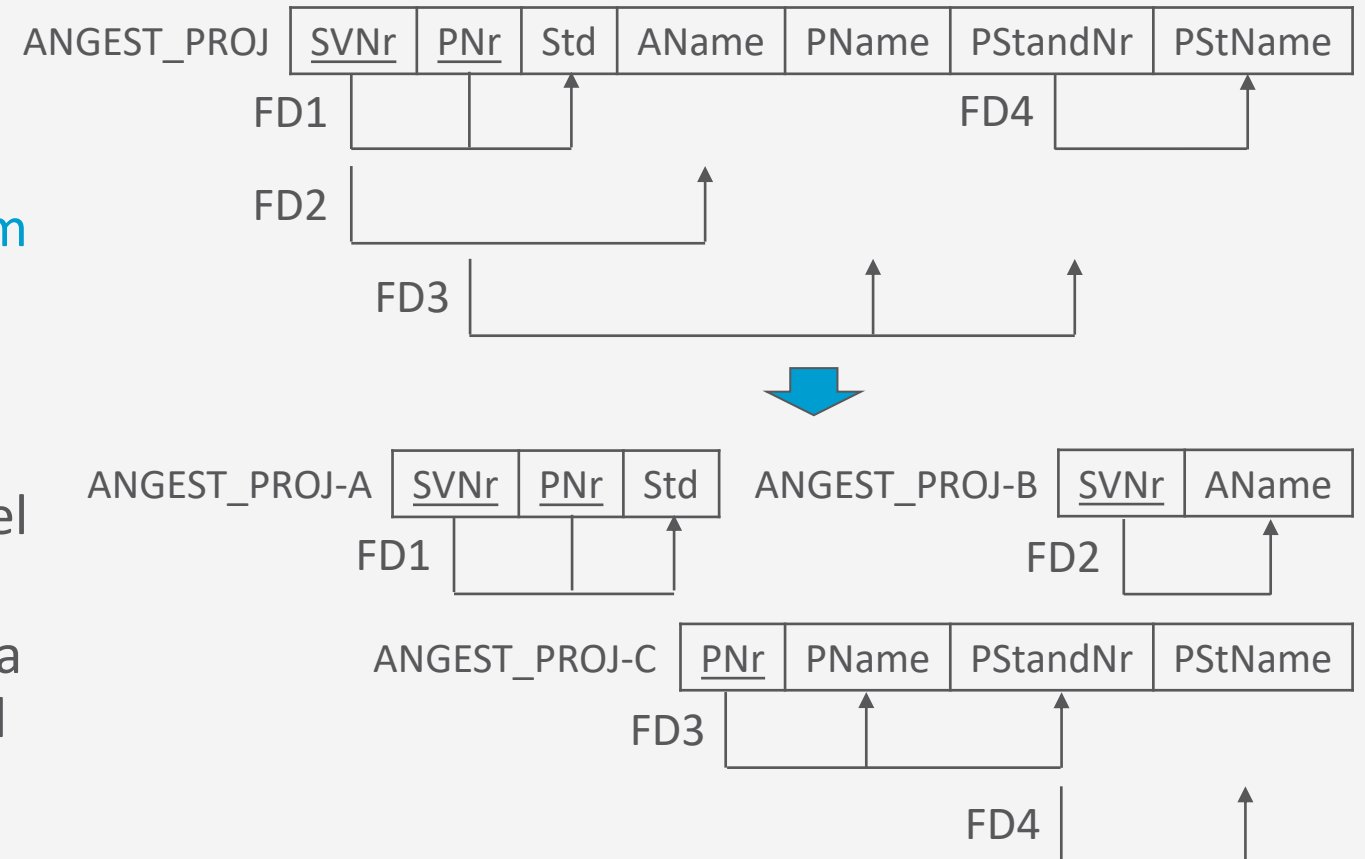
1. Normalform

- Effekt:
 - Keine zusammengesetzten, mengenwertige oder geschachtelten Werte
- Praktischer Nutzen:
 - Erleichtert oder ermöglicht sogar erst Anfragen an die DB durch atomare Einträge
 - Beispiel: Alle Straßen im Ort „Bern“
 - Sonst nicht möglich, da nur Anfragen an alle Straßen in „3000 Bern“, „3018 Bern“ möglich oder mittels aufwendiger Substring-Suche
 - Beispiel: Alle Abteilungen am Standort „Bellaire“
 - Erleichtert, da sonst Suche in Liste für jede Abteilung nötig



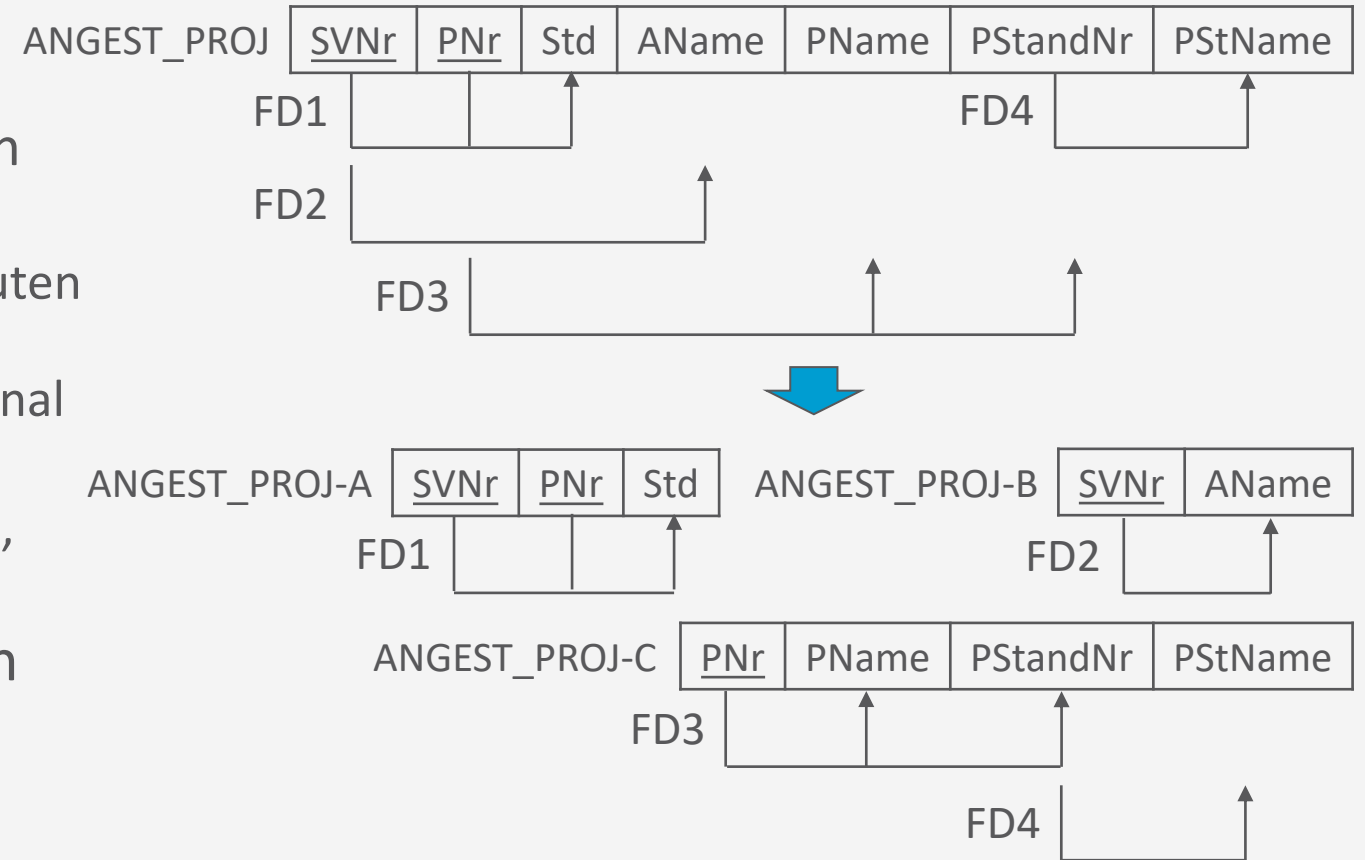
2. Normalform

- Relationenschema R ist in 2. Normalform (2NF), wenn
 - R in 1NF ist und
 - Kein nicht-primäres Attribut von nur einem Teil eines Schlüsselkandidaten abhängt
- Erzeugung:
 - Zerlege das Relationenschema und erstelle ein neues für jeden Teil-Schlüssel mit seinen abhängigen Attributen
 - Erhalte das (restliche) Relationenschema mit dem ursprünglichen Primärschlüssel und Attributen, die von diesem voll funktional abhängig sind



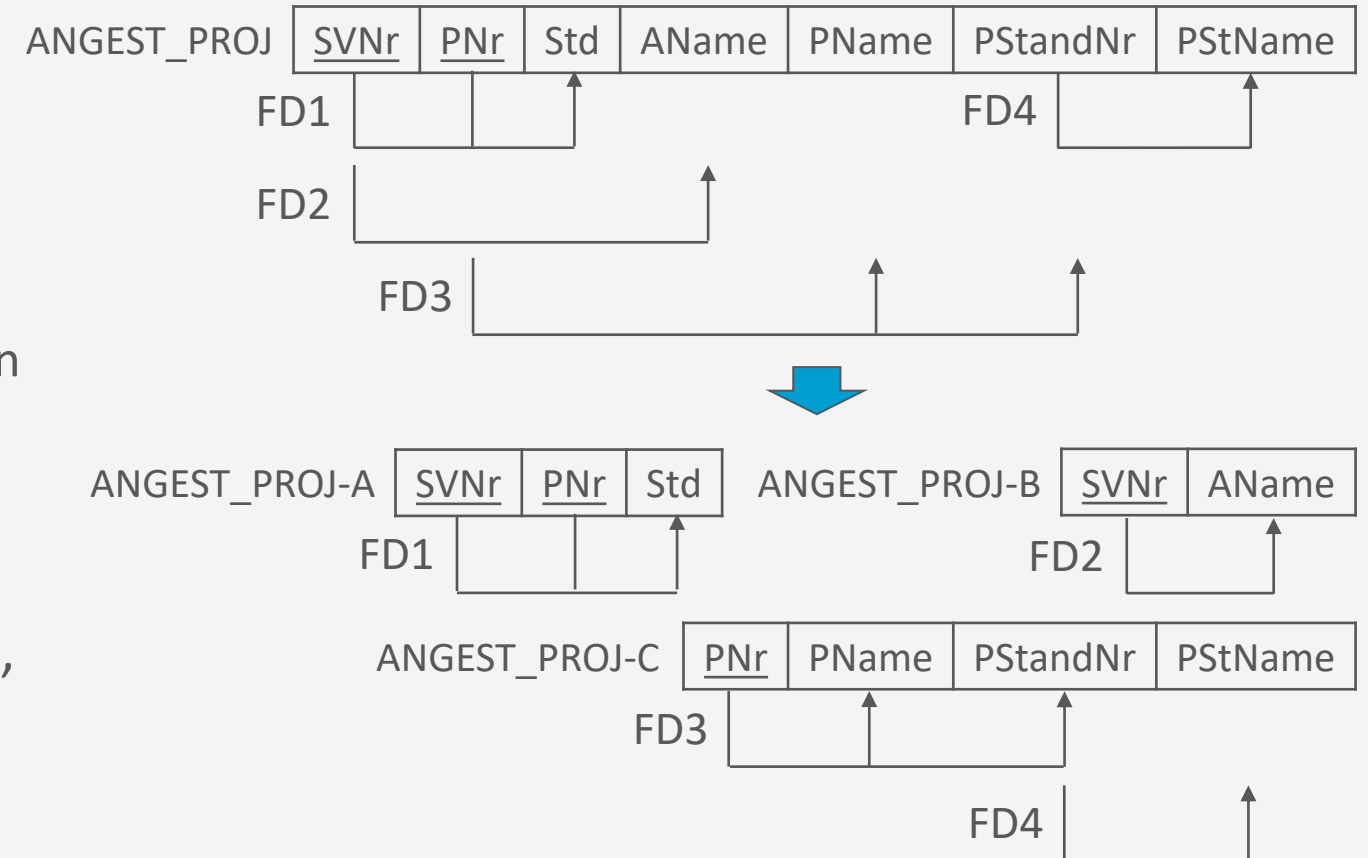
2. Normalform

- Effekt:
 - Alle nicht-primen Attribute sind voll funktional von allen Schlüsselkandidaten abhängig
 - **Voll funktional:** Von allen Schlüsselattributen (und nicht nur einer Teilmenge eines zusammengesetzten Kandidaten) funktional abhängig
 - Heißt auch, wenn nur ein primes Attribut, dann ist eine Relation in 1NF auch in 2NF
- 2NF bei Erzeugung aus ER-Diagramm in der Regel gegeben



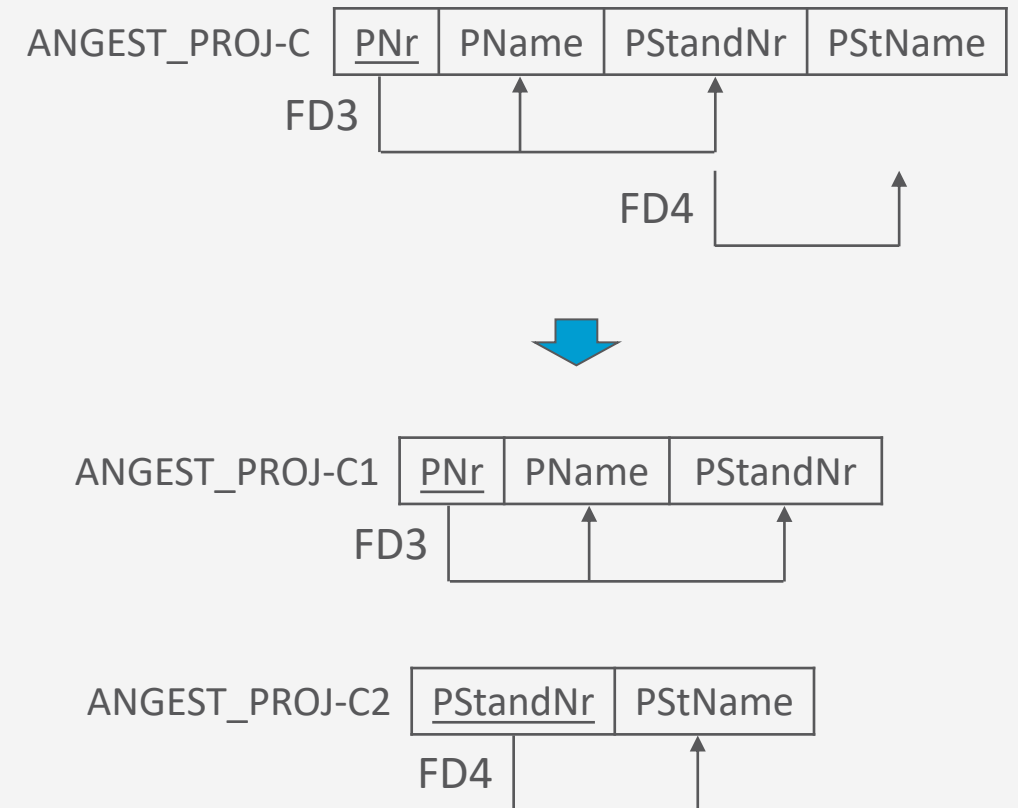
2. Normalform

- Praktischer Nutzen
 - Relationen enthalten logisch zusammengehörige Informationen (*Abgrenzung*)
 - *Reduktion redundanter Werte*
 - Anomalien bei Veränderungen vermeiden
 - Beispiel: *AName* einmal pro *SVNr*
 - Gleiches gilt für Werte pro *PNr*
- Probleme:
 - Immer noch redundante Werte möglich, was zu Anomalien führen kann
 - Beispiel: Standortnamen werden jedes Mal für einen Standort wiederholt



3. Normalform

- Relationenschema R ist in 3. Normalform (3NF), wenn
 - R in 2NF ist und
 - Kein nicht-primales Attribut von R transitiv von einem Kandidatenschlüssel abhängt
 - Keine so genannten *transitiven Abhängigkeiten*
- Erzeugung:
 - Zerlege das Relationenschema und erstelle ein neues, welches das nicht-primale Attribut bzw. die nicht-primären Attribute beinhaltet, die funktional von anderen nicht-primären Attributen bestimmt werden
 - **Synthesealgorithmus**



Synthesealgorithmus

- Ziel: **abhängigkeitswahrende, verlustlose Zerlegung** $D = \{R_1, \dots, R_m\}$ in Bezug auf eine Menge von FDs F eines Relationenschemas R , so dass jede Relation R_i in **3NF** ist
- Input: Universal-Relationenschema R , Menge von FDs F für die Attribute von R
 - Universal-Relationenschema R als eine abstrakte Relation über alle betrachteten Attribute
- Output: Zerlegung $D = \{R_1, \dots, R_m\}$ von R
- 4 Schritte
 1. Bestimme eine kanonische Überdeckung F_C
 2. Erzeuge Relationenschemata für jede FD in F_C
 3. Stelle sicher, dass es ein Schema mit einem Schlüssel für R gibt
 4. Eliminiere doppelte Schemata

Synthesealgorithmus: Schritte

- Input: Universal-Relationenschema R , Menge von FDs F für die Attribute von R
- Output: Zerlegung $D = \{R_1, \dots, R_m\}$ von R
 1. Bestimme eine kanonische Überdeckung F_C für F
 - Linksreduktion, Rechtsreduktion, FDs mit leerem β löschen, FDs mit gleichem α zusammenfassen
 2. Für jede FD $\alpha \rightarrow \beta$ in F_C , erzeuge ein Relationenschema R_α in D mit Attributen $\alpha \cup \beta$
 - Ordne R_α die FDs aus F_C zu, deren Attribute in $\alpha \cup \beta$ sind: $F_\alpha = \{\alpha' \rightarrow \beta' \mid \alpha' \cup \beta' \subseteq \alpha \cup \beta\}$
 - α ist Schlüssel dieses Relationenschemas
 3. Wenn keines der resultierenden Relationenschemata in D einen Schlüssel von R enthält, dann erzeuge ein weiteres Relationenschema in D , das die Attribute enthält, die einen Schlüssel von R bilden: Wähle Kandidatenschlüssel κ und definiere R_κ über κ mit $F_\kappa = \emptyset$
 4. Eliminiere diejenigen Schemata $R_{\alpha'}$ in D , die in einem anderen Schema R_α aus D enthalten sind, d.h. $R_{\alpha'} \subseteq R_\alpha$

Synthesealgorithmus: Beispiel

- Sei unten stehendes Relationenschema R mit funktionalen Abhängigkeiten gegeben

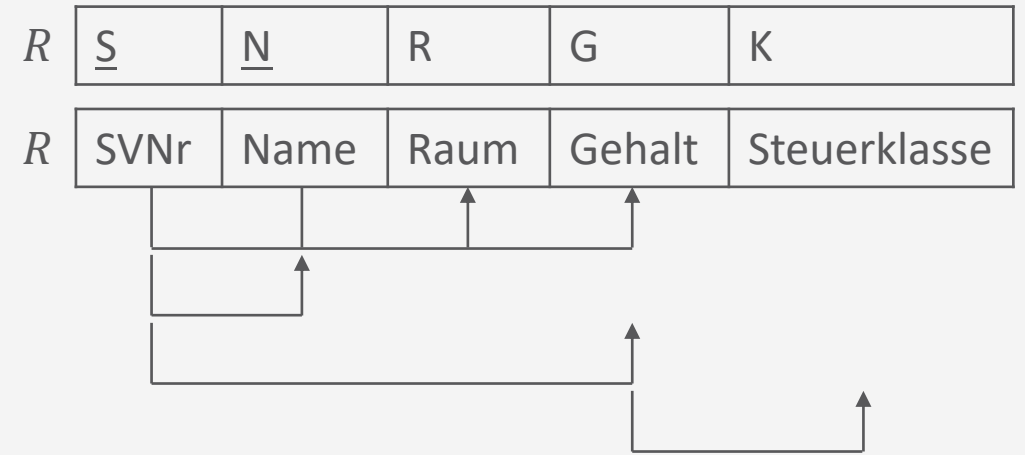
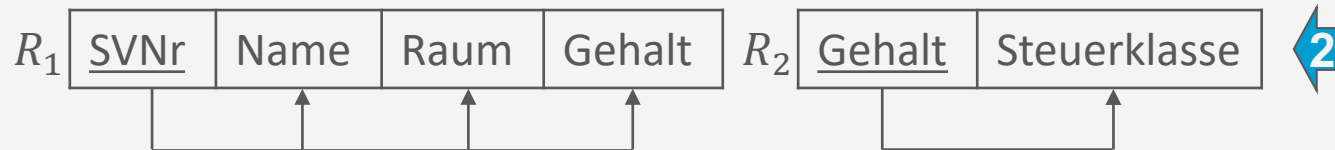
- $F = \{S \rightarrow NG, G \rightarrow K, SN \rightarrow RG\}$

- Bestimme eine kanonische Überdeckung F_C von F

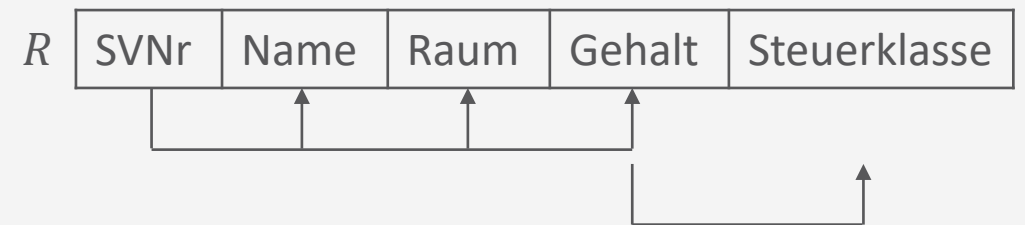
- Ergebnis aus vorherigen Folien:
 $F_C = \{S \rightarrow NRG, G \rightarrow K\}$

- Erzeuge Schemata für die FDs in F_C

- $D = \{R_1, R_2\}$



1



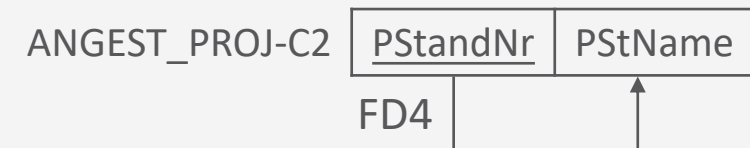
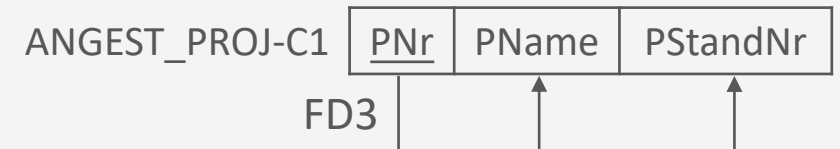
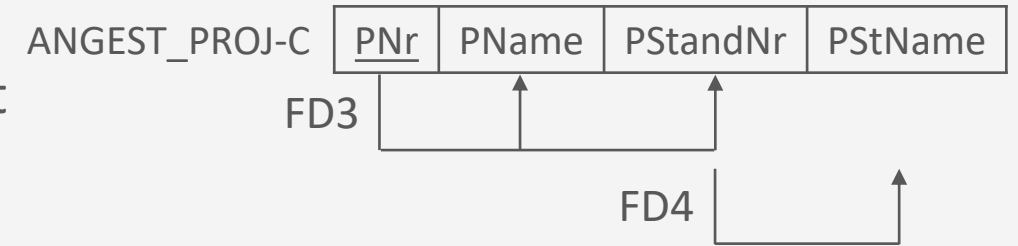
Synthesealgorithmus: Beispiel

- D nach Schritt 2

R_1	<u>SVNr</u>	Name	Raum	Gehalt	R_2	<u>Gehalt</u>	Steuerklasse
-------	-------------	------	------	--------	-------	---------------	--------------
- Schritt 3: Abschließendes Relationenschema bei fehlendem Schlüssel
 - Wenn keines der Relationenschemata in D einen Schlüssel von R enthält, dann erzeuge ein weiteres Relationenschema in D , das Attribute enthält, die einen Schlüssel von R bilden.
 - Betrifft Attribute, die in keiner FD vorkommen
 - Beispiel: Schlüssel von R enthalten (keine Änderungen in D)
- Schritt 4: G reduzieren
 - Eliminiere diejenigen Schemata in D , die in einem anderen Schema aus D enthalten sind
 - Beispiel: R_1 und R_2 sind nicht im jeweils anderen enthalten (keine Änderungen in D)
 - D.h., der Algorithmus terminiert

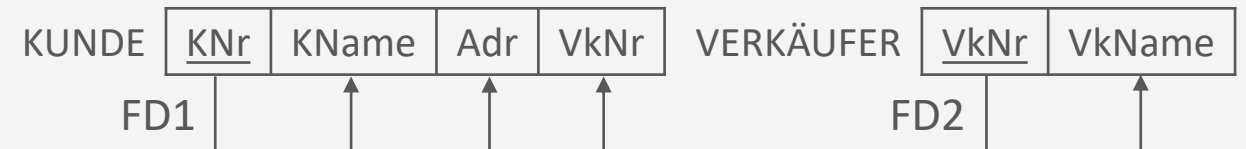
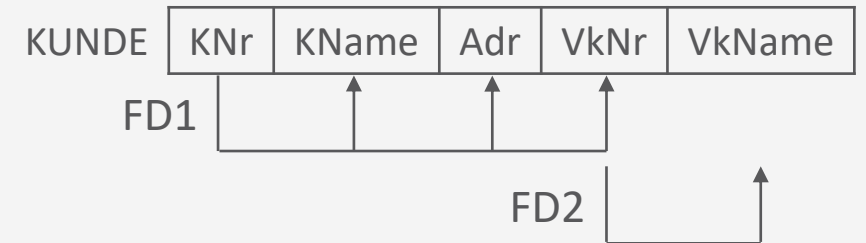
3. Normalform

- Effekt:
 - Es gibt keine anderen Nicht-Schlüsselattribute, von denen ein nicht primes Attribut funktional abhängig ist
- Praktischer Nutzen:
 - Verbliebene thematische Durchmischung behoben (*Abgrenzung*)
 - Weitere *Reduktion redundanter Werte*
 - Weitere Vermeidung von Anomalien
 - Beispiel: Standortnamen nur einmal pro Standortnummer
- Die meisten 3NF-Schemata weisen in der Praxis keine dramatischen Probleme auf
 - Sorgfältige Erstellung, z.B., über ER-Diagramm



3. Normalform: Probleme

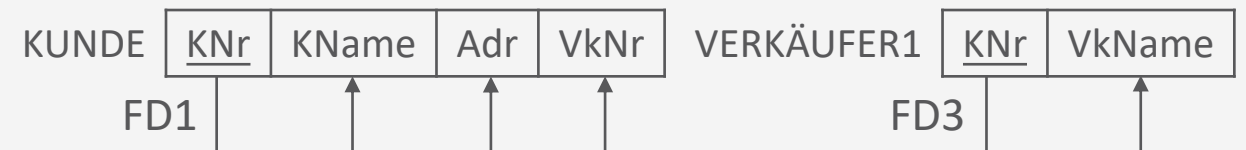
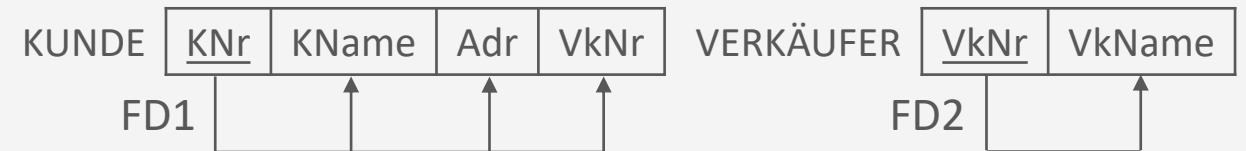
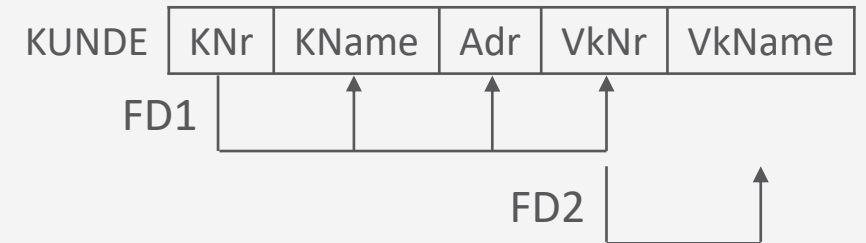
- Immer noch redundante Werte möglich → Anomalien möglich
- Beispiel: $KUNDE(KNr, KName, Adr, VkNr, VkName)$
 - Abhängigkeiten:
 - FD1: $\{KNr\} \rightarrow \{KName, Adr, VkNr\}$
 - FD2: $\{VkNr\} \rightarrow \{VkName\}$
 - In 3NF:
 - $KUNDE(KNr, KName, Adr, VkNr)$
 - $VERKÄUFER(VkNr, VkName)$
 - Problem: Folgende Zerlegung **ebenfalls in 3NF**
 - $KUNDE(KNr, KName, Adr, VkNr)$
 - $VERKÄUFER1(KNr, VkName)$



Was für Anomalien können entstehen?

3. Normalform: Probleme

- Fortsetzung Beispiel:
 - Erzeugt u.a. folgende Anomalien:
 - Änderungsaufwand: Änderung eines $VkName$ zu einer $VkNr$
 - Unvollständiges Einfügen: Einfügen eines neuen Verkäufers mit Nummer und Name, der noch keinen Kunden betreut
 - Ursache: FDs $\{KNr\} \rightarrow \{VkNr\}$ und $\{VkNr\} \rightarrow \{VkName\}$ werden auf zwei Relationen „transitiv“ verteilt
 - IR3: Wenn
 - $\{\{KNr\} \rightarrow \{VkNr\}, \{VkNr\} \rightarrow \{VKName\}\}$,
dann
 - $\{KNr\} \rightarrow \{VKName\}$ (FD3)
 - Reine Erfüllung der NF kein gutes Maß



3. Normalform: Probleme

- Nicht-triviale Abhängigkeiten zwischen Attributen von Schlüsselkandidaten → Erzeugen ebenfalls Redundanzen
- Beispiel: $RECHNUNG(RNr, KostenArt, KANr, Anzahl)$
 - Abhängigkeit zwischen $KostenArt, KANr$ (1:1-Beziehung)
 - Schlüsselkandidaten: $\{RNr, KostenArt\}, \{RNr, KANr\}$
 - In 3NF, da
 - In 2NF: jedes nicht-prime Attribut ($Anzahl$) ist voll funktional abhängig von allen Attributen der jeweiligen Schlüsselkandidaten
 - Keine transitiven Abhängigkeiten, da kein weiteres nicht-primales Attribut vorhanden
 - Aber:
 - Redundanzen durch Abhängigkeit zwischen $KostenArt, KANr$ → Anomalien möglich, z.B. wenn man eine Nummer zu einer Kostenart ändert

RECHNUNG	RNr	KostenArt	KANr	Anzahl
----------	-----	-----------	------	--------

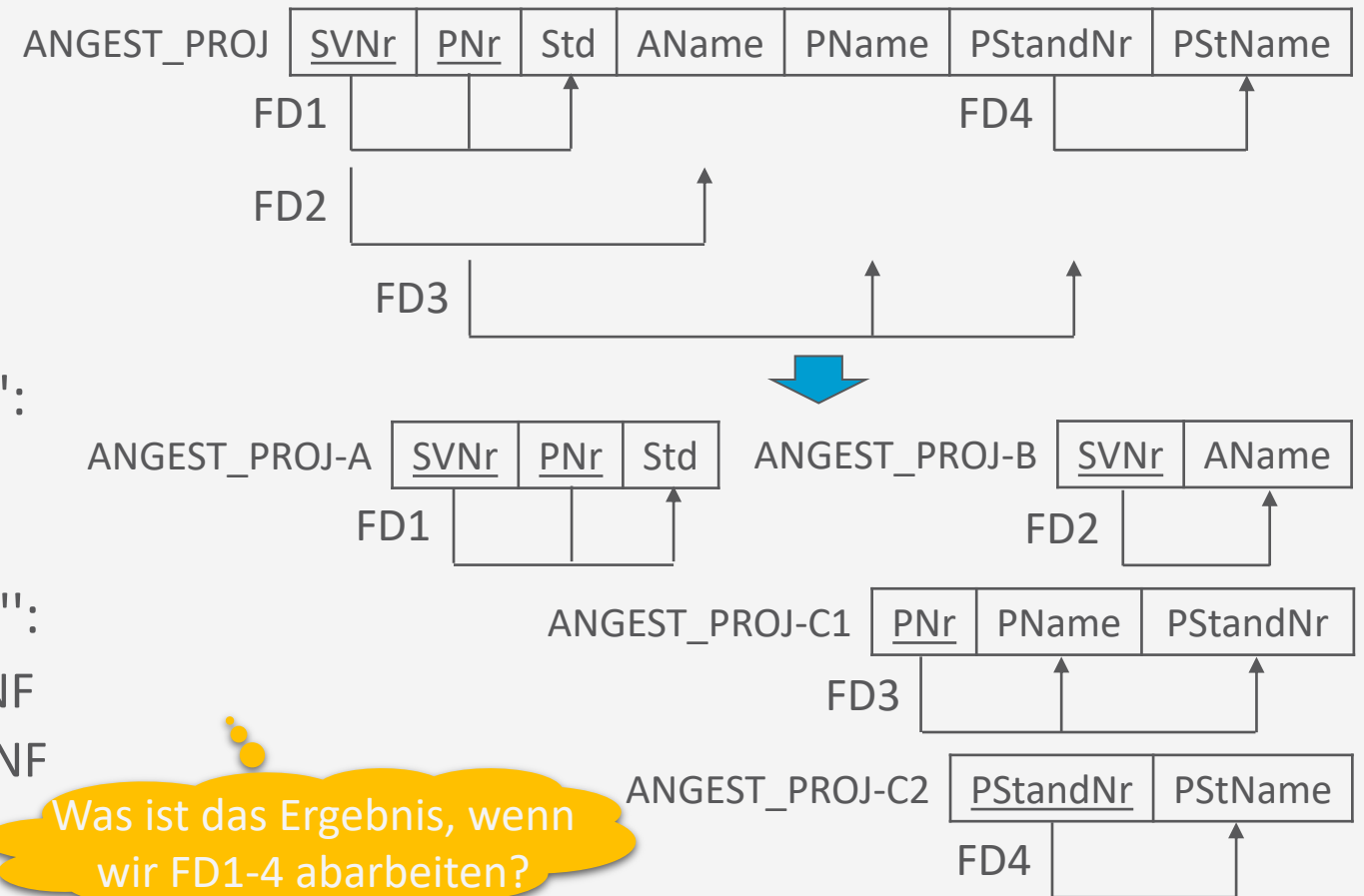
In 3NF ist es möglich, dass ein Teil eines (zusammengesetzten) Schlüsselkandidaten funktional abhängig ist von einem Teil eines anderen Schlüsselkandidaten.

Zerlegungsalgorithmus

- Eingabe: Relationenschema R und Menge von zugehörigen FDs F
- Ausgabe: Zerlegung $D = \{R_1, \dots, R_n\}$, wobei jedes R_i in BCNF ist
- Starte mit $D = \{R\}$
- Solange es noch ein Relationenschema R_i in D gibt, das nicht in BCNF ist:
 - Finde eine FD $\alpha \rightarrow \beta$, die dem R_i zugehörig ist, mit
 1. $\alpha \cap \beta = \emptyset$
 2. $\neg(\alpha \rightarrow R_i)$
 - Man sollte $\alpha \rightarrow \beta$ so wählen, dass β alle von α funktional abhängigen Attribute $B \in (R_i - \alpha)$ enthält, damit der Dekompositionsalgorithmus möglichst schnell terminiert
 - Zerlege R_i in $R_{i1} \leftarrow \alpha \cup \beta$ und $R_{i2} \leftarrow R_i - \beta$
 - Entferne R_i aus D und füge R_{i1} und R_{i2} ein, also
 - $D \leftarrow (D - \{R_i\}) \cup \{R_{i1}, R_{i2}\}$

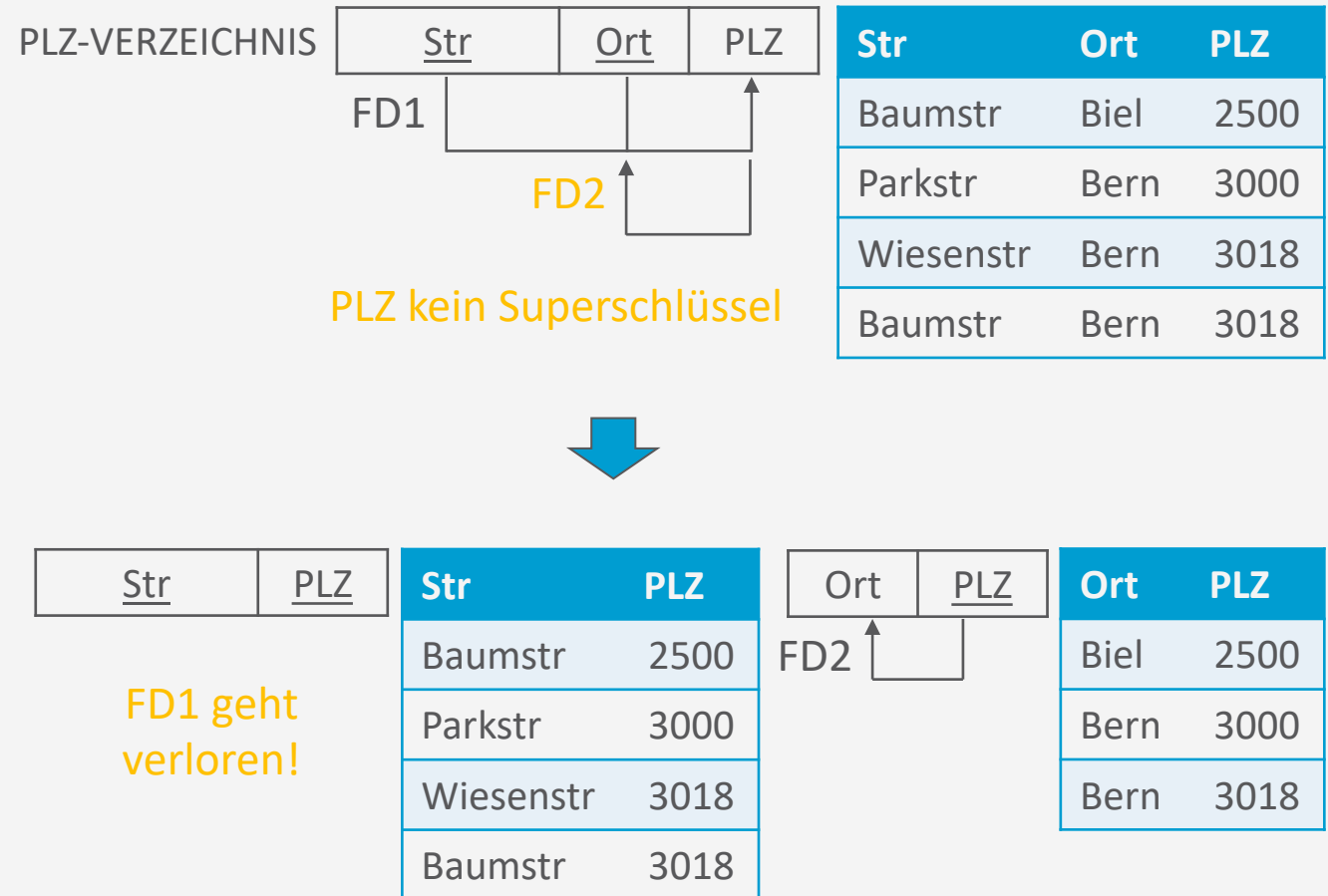
Beispiele

- Zerlegung ANGEST_PROJ
 - FD4 erfüllt $\neg(\alpha \rightarrow R_i)$:
 - ANGEST_PROJ-C2 \rightarrow in BCNF
 - ANGEST_PROJ'(SVNr, PNr, Std, AName, Pname, PStandNr)
 - FD3 erfüllt $\neg(\alpha \rightarrow R_i)$ in ANGEST_PROJ':
 - ANGEST_PROJ-C1 \rightarrow in BCNF
 - ANGEST_PROJ''(SVNr, PNr, Std, AName)
 - FD2 erfüllt $\neg(\alpha \rightarrow R_i)$ in ANGEST_PROJ'':
 - ANGEST_PROJ-B(SVNr, AName) \rightarrow in BCNF
 - ANGEST_PROJ-A(SVNr, PNr, Std) \rightarrow in BCNF



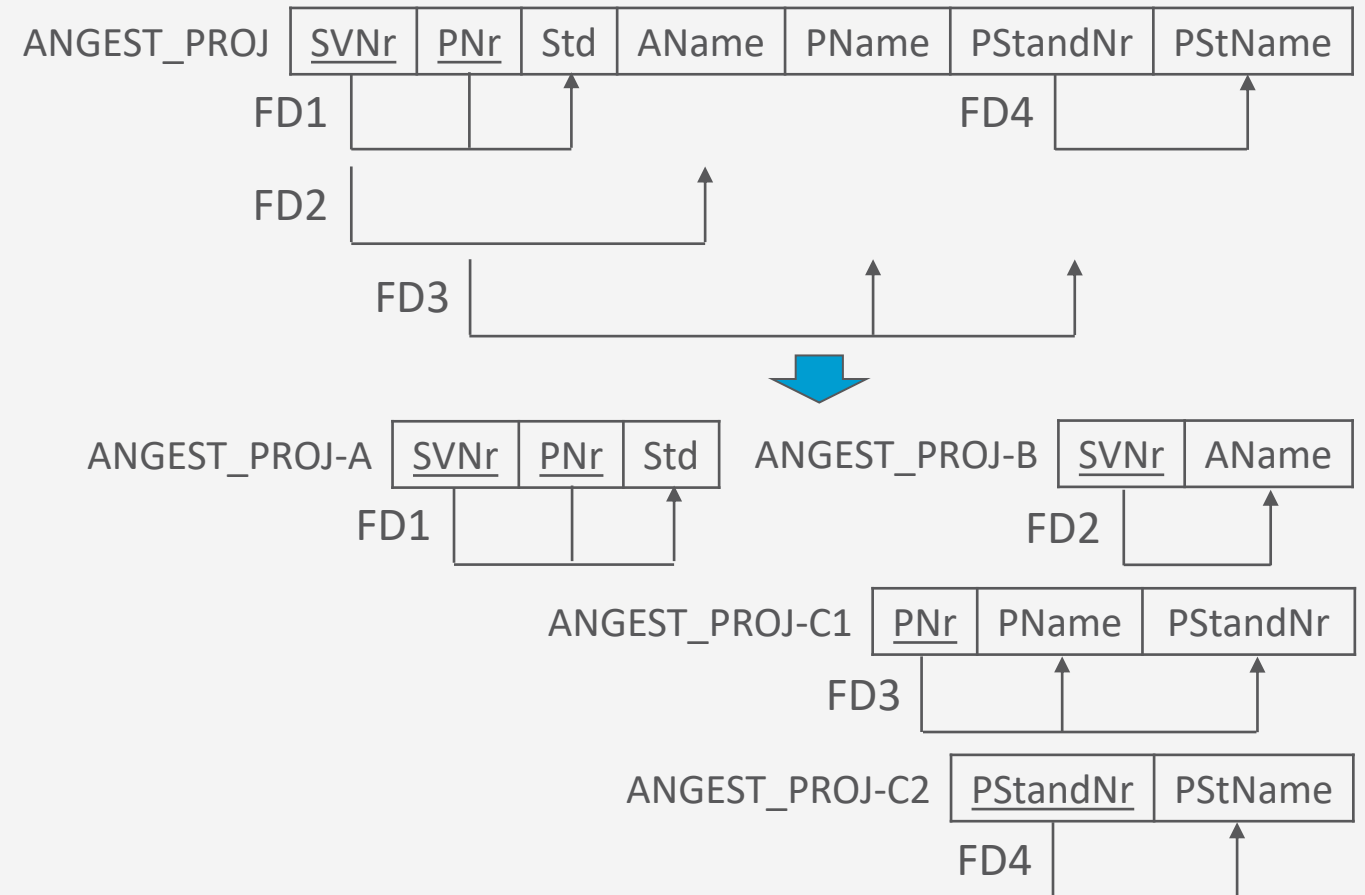
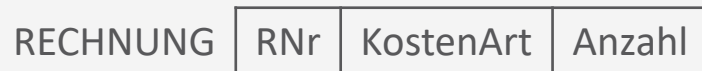
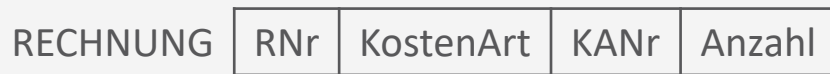
Boyce-Codd-Normalform (BCNF)

- Relation lässt sich immer in BCNF bringen
 - Dann keine Redundanzen mehr
 - Verlustlos
- Aber: Nicht jedes 3NF Schema in BCNF überführbar bei gleichzeitiger Wahrung der Abhängigkeiten
 - Damit keine effiziente Überprüfung der Abhängigkeiten möglich
 - Beispiel: Man erhält nicht mehr ohne Join die PLZ zu Straße und Ort



BCNF und 3NF

- Unterschied zwischen 3NF und BCNF nur bei mehreren Schlüsselkandidaten mit überlappenden Attributen
- Relation in BCNF hat garantiert keine Redundanzen
 - Im Gegensatz zu Relation in 3NF



Höhere Normalformen

- 4. Normalform (4NF): Keine mehrwertigen Abhängigkeiten (*multi-valued dependencies*, MVDs)
 - Falls in einem Relationenschema $R(A_1, \dots, A_n)$ wenigstens zwei 1:n-Beziehungen der Form $A_i:A_j$ und $A_i:A_k$ existieren, bei denen A_j und A_k „unabhängig“ sind, dann kann eine MVD entstehen
 - zu zerlegen für 4NF
 - Beispiel
 - Ein Angestellter arbeitet an Projekten und hat Angehörige. Er kann an mehreren Projekten arbeiten und ebenfalls mehrere Angehörige haben. Projekte und Angehörige sind aber voneinander unabhängig.
 - In *ANGEST* sind die beiden 1:n-Beziehungen in einer Relation, bei *ANGEST_PROJ*, *ANGEST_ANG* auf zwei Relationen verteilt

ANGEST	<u>AName</u>	<u>PName</u>	<u>AName</u>
	Smith	P1	John
	Smith	P2	Anna
	Smith	P1	Anna
	Smith	P2	John



ANGEST_PROJ	<u>AName</u>	<u>PName</u>
	Smith	P1
	Smith	P2

ANGEST_ANG	<u>AName</u>	<u>AName</u>
	Smith	John
	Smith	Anna

Höhere Normalformen

- 5. Normalform (5NF)
 - Es gibt Fälle, in denen eine nicht-additive JOIN-Zerlegung eines Schemas R in zwei Relationenschemata nicht möglich ist, aber in drei oder mehr
 - Beispiel:
 - Zerlegung von R in zwei Schemata nicht möglich (unechte Tupel)
 - Nicht-additive JOIN-Zerlegung in R_1 , R_2 und R_3 möglich
 - Man sagt auch: „JOIN-Abhängigkeit“ verhindern

R	<u>Repräsentant</u>	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford	PKW
	Schmidt	Ford	LKW
	Schmidt	VW	Transporter
	Müller	Porsche	PKW
	Müller	Ford	PKW



R_1	<u>Repräsentant</u>	<u>Firma</u>	R_2	<u>Repräsentant</u>	<u>Produkt</u>	R_3	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford		Schmidt	PKW		Ford	PKW
	Schmidt	VW		Schmidt	LKW		Ford	LKW
	Müller	Porsche		Schmidt	Transporter		VW	Transporter
	Müller	Ford		Müller	PKW		Porsche	PKW

Die 5. Normalform – Motivation

- Beispiel (Test 1): JOIN über R_1 und R_2 (Join-Prädikat Repräsentant)

R	<u>Repräsentant</u>	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford	PKW
	Schmidt	Ford	LKW
*	Schmidt	Ford	Transporter
*	Schmidt	VW	PKW
*	Schmidt	VW	LKW
	Schmidt	VW	Transporter
	Müller	Porsche	PKW
	Müller	Ford	PKW

* **unechte Tupel**

$R_1 * R_2$

R_1	<u>Repräsentant</u>	<u>Firma</u>	R_2	<u>Repräsentant</u>	<u>Produkt</u>	R_3	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford		Schmidt	PKW		Ford	PKW
	Schmidt	VW		Schmidt	LKW		Ford	LKW
	Müller	Porsche		Schmidt	Transporter		VW	Transporter
	Müller	Ford		Müller	PKW		Porsche	PKW

Die 5. Normalform – Motivation

- Beispiel (Test 2): JOIN über R_1 und R_3 (Join-Prädikat Firma)

R	<u>Repräsentant</u>	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford	PKW
	Schmidt	Ford	LKW
	Schmidt	VW	Transporter
	Müller	Porsche	PKW
	Müller	Ford	PKW
*	Müller	Ford	LKW

* **unechte Tupel**

$R_1 * R_3$

R ₁	<u>Repräsentant</u>	<u>Firma</u>	R ₂	<u>Repräsentant</u>	<u>Produkt</u>	R ₃	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford		Schmidt	PKW		Ford	PKW
	Schmidt	VW		Schmidt	LKW		Ford	LKW
	Müller	Porsche		Schmidt	Transporter		VW	Transporter
	Müller	Ford		Müller	PKW		Porsche	PKW

Die 5. Normalform – Motivation

- Beispiel (Test 3): JOIN über R_2 und R_3 (Join-Prädikat Produkt)

R	<u>Repräsentant</u>	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford	PKW
*	Schmidt	Porsche	PKW
	Schmidt	Ford	LKW
	Schmidt	VW	Transporter
	Müller	Porsche	PKW
	Müller	Ford	PKW

* **unechte Tupel**

$R_2 * R_3$	R_1	<u>Repräsentant</u>	<u>Firma</u>	R_2	<u>Repräsentant</u>	<u>Produkt</u>	R_3	<u>Firma</u>	<u>Produkt</u>
		Schmidt	Ford	Schmidt	PKW	Ford	PKW		
		Schmidt	VW	Schmidt	LKW	Ford	LKW		
		Müller	Porsche	Schmidt	Transporter	VW	Transporter		
		Müller	Ford	Müller	PKW	Porsche	PKW		

Die 5. Normalform – Motivation

- Beispiel (Test 4): JOIN über R_1 , R_2 und R_3

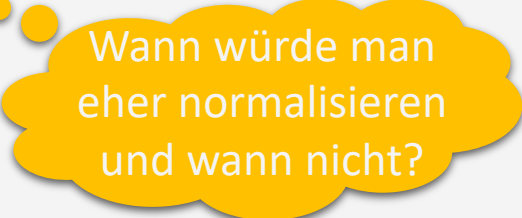
R	<u>Repräsentant</u>	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford	PKW
	Schmidt	Ford	LKW
	Schmidt	VW	Transporter
	Müller	Porsche	PKW
	Müller	Ford	PKW

 $R_1 * R_2 * R_3$

R_1	<u>Repräsentant</u>	<u>Firma</u>	R_2	<u>Repräsentant</u>	<u>Produkt</u>	R_3	<u>Firma</u>	<u>Produkt</u>
	Schmidt	Ford		Schmidt	PKW		Ford	PKW
	Schmidt	VW		Schmidt	LKW		Ford	LKW
	Müller	Porsche		Schmidt	Transporter		VW	Transporter
	Müller	Ford		Müller	PKW		Porsche	PKW

Normalisierung und die Praxis

- Aus Performance-Gründen manchmal keine Normalisierung
- Gründe gegen Normalisierung, z.B.
 - Zerlegung kann häufige JOINS erfordern
 - Wenn Dinge immer wieder vorkommen, ist es vielleicht sinnvoll die JOIN Tabelle zu speichern, auch wenn es Redundanzen gibt
 - Wenn es kaum Änderungen am Datensatz gibt, dann sind Redundanzen nicht so tragisch
- Gründe für Normalisierung, z.B.
 - Häufiger Änderungen
 - Abwägung zwischen Redundanzfreiheit und Abhängigkeitswahrung, sollte nur eins von beiden gehen



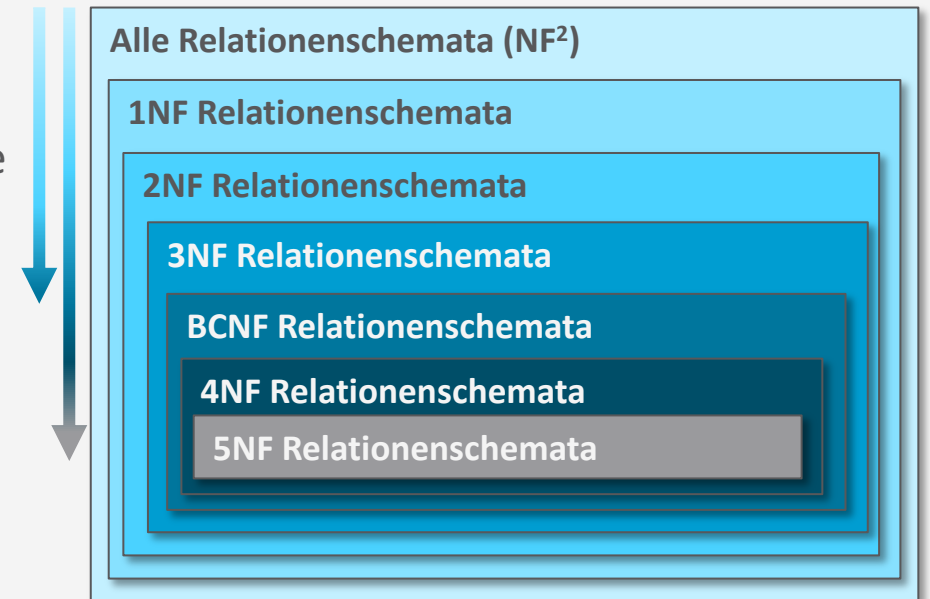
Wann würde man eher normalisieren und wann nicht?

Zwischenzusammenfassung

- Korrektheitskriterien für Zerlegungen: Verlustlose Zerlegung, Abhängigkeitswahrung
 - Dazu: Vermeidung von Redundanzen
- Normalformen
 - (NF²), 1NF, 2NF, 3NF, BCNF, 4NF, 5NF
 - Synthesealgorithmus liefert verlustfreie, abhängigkeitswahrende Zerlegung mit Schemata in 3NF; Redundanzen möglich
 - Zerlegungsalgorithmus liefert verlustfreie Zerlegung ohne Redundanzen mit Schemata in BCNF, aber nicht immer abhängigkeitswährend

Abhängigkeitserhaltende
Zerlegung
(bis 3NF)

Verlustlose Zerlegung
(bis 5NF)



Überblick: 4. Datenbankentwurf

A. Funktionale Abhängigkeiten

- Definition, Schlüssel, Ableitungen
- Attributhülle, kanonische Überdeckung

B. Normalformen

- Zerlegung von Relationen
- Exkurs: NF^2 ; 1NF, 2NF, 3NF, BCNF, 4NF, 5NF
- Synthesealgorithmus, Zerlegungsalgorithmus

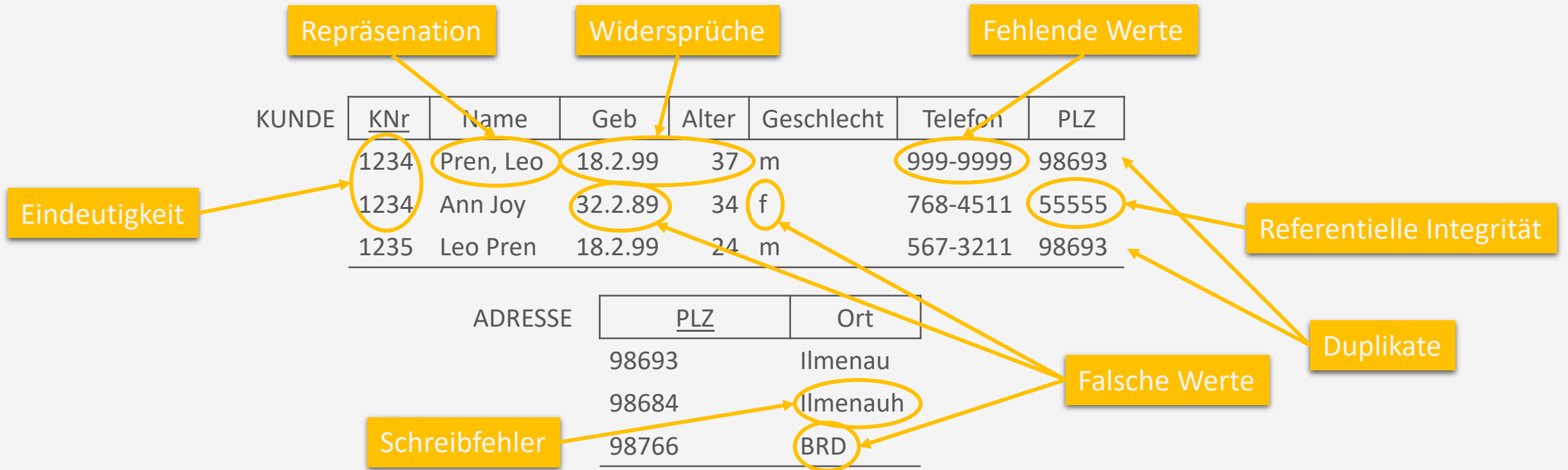
C. Datenqualität

- Datenqualitätsprobleme, Dimensionen der Datenqualität

Problemfeld Datenqualität

- Normalformen zielen auf die **strukturelle Qualität** von Relationenschemata
 - „Intension“
- Datenqualität umfasst Beiträge zur **Verbesserung der konkreten „inhaltlichen“ Dateninstanzen** auf der Schemaebene
 - „Extension“
- Der Aspekt der Datenqualität ist als problematisch einzuschätzen, wenn Daten vor dem Hintergrund einer Anwendungssituation, z.B.
 - Nicht die angenommene Bedeutung haben
 - Nicht der Spezifikation entsprechen
 - Unverständlich sind

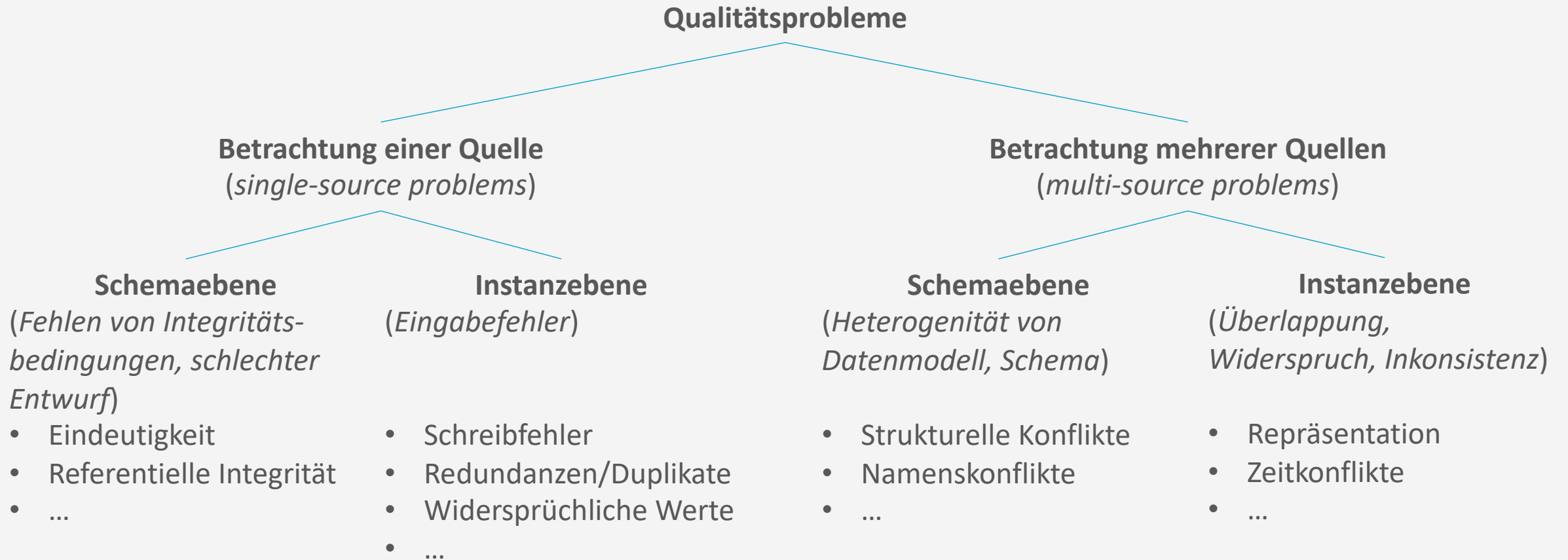
Beispiele für mindere Datenqualität



Ursachen für Datenqualitätsprobleme

- Ursachen liegen auf verschiedenen Ebenen, u.a. bei der
 - Datenproduktion, z.B.
 - Verschiedene Quellen repräsentieren gleiche Realwelt-Objekte in unterschiedlicher Form
 - Datenerfassung unterliegt „subjektiven Einflüssen“
 - Systematische Probleme bei Datenerfassung (Messung, Kodierung etc.)
 - Datenspeicherung, z.B.
 - Unterschiedliche oder ungeeignete Formate
 - Datennutzung, z.B.
 - Unzureichende Analyse- und Verarbeitungsmöglichkeiten
 - Veränderung der Nutzungsbedürfnisse
 - Sicherheits- und Zugriffsprobleme

Arten von Datenqualitätsproblemen



Dimensionen der Datenqualität

- Datenqualität wird anhand verschiedener Dimensionen beurteilt, z.B.
 - **Vollständigkeit**: Verhältnis tatsächlicher Werte zu gespeicherten Werten, u.a.
 - Wertebelegungen verschieden von Null
 - Repräsentation aller in der Realwelt vorkommenden Objekte
 - **Genauigkeit**: Verhältnis der Anzahl der korrekten Werte zur Gesamtanzahl, d.h. prozentualer Anteil an Daten ohne Datenfehler
 - Umfang, in dem Attributwerte im jeweils „optimalen“ Detaillierungsgrad vorliegen
 - Nähe eines Wertes zum korrekten Wert innerhalb der Realwelt
 - **Zeitnähe**: Aktualität, in der Attributwerte dem sich dynamisch ändernden Realwelt-Zustand entsprechen
 - Alter: Zeit seit dem Erfassen / Laden der Daten
 - Volatilität: Häufigkeit der Änderungen

Dimensionen der Datenqualität (Fortsetzung)

- Weitere Dimensionen, u.a.
 - **Relevanz**: Grad, in dem der Informationsgehalt den Nutzungsbedürfnissen entspricht
 - **Verständlichkeit**: Grad, in dem Daten in Inhalt und Struktur mit der „Vorstellungswelt“ der nutzenden Personen übereinstimmen
 - **Konsistenz**: Grad, in dem Daten frei von logischen Widersprüchen sind
 - Integritätsbedingungen, Geschäftsregeln, ...
 - **Verfügbarkeit**: Grad, in dem Daten für nutzende Personen in einem bestimmten Zeitraum nutzbar sind
 - **Glaubwürdigkeit**: Grad, in dem Daten von nutzenden Personen als korrekt akzeptiert werden
 - **Kosten**: Preis für Datenzugriff, Anfrage, Datenübertragung, ...

Zwischenzusammenfassung

- Normalformen: strukturelle Qualität, Datenqualität: Qualität der Dateninstanzen
- Ursachen von Datenqualitätsproblemen
 - Verschiedene Quellen, subjektive oder systematische Einflüsse bei der Datenerfassung
 - Unterschiedliche / falsche Formate
 - Veränderte Anforderungen, Sicherheits- und Zugriffsprobleme
- Arten von Datenqualitätsproblemen
 - Eindeutigkeit, referentielle Integrität, Schreibfehler, Redundanzen / Duplikate, Widersprüche
 - Namenskonflikte, strukturelle Konflikte, Repräsentation
- Dimension Beurteilung
 - Vollständigkeit, Genauigkeit, Zeitnähe, Relevanz, Verständlichkeit, Konsistenz, Verfügbarkeit, Glaubwürdigkeit, Kosten

Überblick: 4. Datenbankentwurf

A. Funktionale Abhängigkeiten

- Definition, Schlüssel, Ableitungen
- Attributhülle, kanonische Überdeckung

B. Normalformen

- Zerlegung von Relationen
- Exkurs: NF^2 ; 1NF, 2NF, 3NF, BCNF, 4NF, 5NF
- Synthesealgorithmus, Zerlegungsalgorithmus

C. Datenqualität

- Datenqualitätsprobleme, Dimensionen der Datenqualität

→ Structured Query Language (SQL)