Aufbau von Kapitel 5

- Zunächst werden Qualitätskriterien für "gute" Datenbanken erarbeitet, wobei es sich primär um **informelle Richtlinien** für den Datenbankentwurf handelt.
- Kapitel 5.2 greift nochmals den Begriff der **semantischen Integritätsbedingungen** auf und verdeutlicht diesen anhand einiger Beispiele.
- Anschließend werden formale Kriterien zur Beurteilung der Qualität des Datenbankentwurfs besprochen. Schwerpunkt sind hierbei die sog. **funktionalen Abhängigkeiten** sowie 2NF, 3NF und BCNF.
- In Abschnitt 5.4 werden dann noch **mehrwertige Abhängigkeiten** und, darauf aufbauend, die 4. Normalform behandelt.

Semantik von Relationen und Attributen

- Bereits bei E/R-Diagrammen wurden Entities und die zugehörigen Attribute durch Abstraktion zu Entitytypen gruppiert.
- Auch beim relationalen Datenbankentwurf werden mit einer Relation bestimmte Fakten und Aussagen assoziiert, d.h. es wird eine bestimmte Bedeutung bzw. Semantik unterstellt, die angibt, in welcher Beziehung die Relationen zueinander stehen und wie die einzelnen Attributwerte eines Tupels zu interpretieren sind.
- Allgemein gilt: Je einfacher die Semantik einer Relation ist, desto verständlicher ist der Entwurf des Relationenschemas insgesamt.

Relationales DB-Schema Firma (vgl. Elm-02, Seite 502 ff.)

Angestellte	**AngNr**	Name	Vorname	Beruf	AbtNr
Abteilungen	**AbtNr**	AbtBez	AngNrAbtL		
Projekte	**ProNr**	ProBez	ProBudget	AngNrPL	
Mitarbeit	**AngNr**	**ProNr**	Umfang		

Richtlinie 1 (vgl. Elm-02, Seite 502)

Ein Relationenschema sollte so entworfen werden, dass sich seine Bedeutung leicht erfassen lässt.

Kombiniere daher keine Attribute aus mehreren Entity- und /oder Beziehungstypen zu einer einzigen Relation. Durch diese "Mischung" von Attributen wird der Entwurf häufig semantisch unklar und für Dritte schwer verständlich .

Wenn dagegen eine Relation genau einem Entity- oder Beziehungstyp der realen Welt entspricht, erhöht das die Lesbarkeit und Verständlichkeit des gesamten Schemas.

Wie beurteilen Sie dieses Schema?

Firma:

AngNr	Name	Vorname	Beruf	AbtNr	AbtBez	AngNrAbtL
ProNr	ProBez	ProBudget	AngNrPL	Umfang		

- Eine klare Semantik gemäß Richtlinie 1 kann man als gegeben unterstellen, weil hier der Entity-Typ Firma als Ganzes abgebildet wird und dies durchaus nachvollziehar ist.
- Fragen
  - Schlüssel der Relation Firma?
  - Gibt es Nachteile gegenüber dem zuvor beschriebenen Schema mit 4 Tabellen und, falls ja, welche?

Anomalien (nach Codd)

- **Einfügeanomalie (insertion anomaly)**
- **Löschanomalie (deletion anomaly)**
- **Modifikationsanomalie (update anomaly)**

Richtlinie 2 (vgl. Elm-02, Seite 506)

Ein Relationenschema ist so zu entwerfen, dass keine Einfüge-, Lösch- und Update-Anomalien auftreten.

Alle Programme, welche die Datenbank aktualisieren, müssen sicherstellen, dass mit den Anomalien fehlerfrei umgegangen wird und die Datenbank zu jedem Zeitpunkt ein korrektes Abbild der Realwelt darstellt.

Überlegungen zu Nullwerte in Tupeln

- Nullwerte lassen grundsätzlich mehrere Interpretationen zu:
  - Das Attribut ist auf dieses Tupel nicht anwendbar.
  - Der Attributwert ist für dieses Tupel unbekannt.
  - Der Attributwert ist für dieses Tupel zwar bekannt, aber noch nicht erfasst.
- Nullwerte verschwenden Speicherplatz!
- Die Auswertung von Aggregatsfunktionen wie AVG oder SUM führt bei Nullwerten möglicherweise zu Missverständnissen.
- Beispiel: Wenn nur 10% aller Angestellten ein eigenes Büro haben (der Rest arbeitet z.B. in der Produktion), gibt es keine besonders gute Recht-fertigung, die Büronummer in der Relation Angestellte unterzubringen.

Richtlinie 3 (vgl. Elm-02, Seite 507)

Ein Relationenschema sollte so entworfen werden, dass Nullwerte bei Attributen einer Relation nur ausnahmsweise auftreten und nicht auf die Mehrheit aller Tupel zutreffen.

Wenn absehbar ist, dass beim Einfügen von Tupeln in eine Relation sehr häufig Nullwerte vorkommen, ist das Schema zu verbessern.

Betrachten wir ein weiteres Beispiel:

PersNr	Name	Rang	Raum	Telefon	VorlNr	Titel	SWS
1571	Keppler	C4	122	445623	2016	Aerostatik	2
1887	Schrödinger	C2	217	547864	2020	Amorphe Stoffe	2
1901	Heisenberg	C4	141	698472	2009	Die Unschärferelation	3
1571	Keppler	C4	122	445623	2007	Theoretische Mechanik I	2

- Welche Probleme fallen Ihnen hier spontan auf?

Schlussfolgerung

- Viele kleine Relationen scheinen deutlich günstiger zu sein als eine oder wenige große Relation(en).
- **Fazit: Also alle Relationen möglichst fein zerlegen?**
- Frage: Kann man eine Relation überhaupt beliebig zerlegen?

Dekomposition von Relationen

- Die zuvor dargestellten Probleme sind natürlich darauf zurückzuführen, dass logisch nicht "zusammenpassende" Informationen in einer (großen) Relation gebündelt wurden.
- Auf den ersten Blick wäre es also naheliegend, die Probleme zu vermeiden, indem man Relationen möglichst fein zerlegt.
- **Fazit: Also alle Relationen grundsätzlich zerlegen?**
- Hier stellt sich die Frage, ob die Zerlegung einer großen Relation in mehrere kleinere Relationen nicht ebenfalls mit Problemen verbunden ist.

Übungsaufgabe

- Gegeben sei die folgende (praxisorientierte) Relation:

Restaurant	Gast	Getränk
Bella Italia	Müller	Wein
Bella Italia	Schmitt	Bier
San Remo	Müller	Cola

- Aufgabe: Bitte zerlegen Sie diese Relation durch Projektion in die Relationen *besucht* $\pi_{\text{Restaurant,Gast}}$ und *trinkt* $\pi_{\text{Gast,Getränk}}$
- Bilden Sie nun den Join *besucht* $\bowtie$ *trinkt* und vergleichen Sie das Ergebnis mit der Ausgangsrelation.

Richtlinie 4 (vgl. Elm-02, Seite 509)

Ein Relationenschema sollte so entworfen werden, dass Relationen durch einen Join mit Gleichheitsbedingung auf Attributen, die weder Primär- noch Fremdschlüssel sind, so zusammengeführt werden können, dass keine unechten Tupel* entstehen.

Es sollte aber auch vermieden werden, dass Relationen außer den Primär- bzw. Fremdschlüsseln gleichlautende Attribute haben.

*Anmerkung: Zusätzliche Tupel, die bei einem Join entstehen und in der Ursprungsrelation überhaupt nicht enthalten waren, werden als unechte Tupel bezeichnet, weil sie falsche bzw. ungültige Sachverhalte darstellen.

Zusammenfassung der Entwurfsrichtlinien 1 - 4

- In diesem Abschnitt wurden informelle Kriterien zur qualitativen Beurteilung von relationalen Datenbankschemata besprochen. Kennzeichnend ist hierbei allerdings, dass kein spezielles Analyseverfahren zum Erkennen von problematischen Schemata existiert.

- Zusammenfassend sind folgende Punkte zu berücksichtigen:
  - Relationsschema klar und verständlich aufbauen und im Modell möglichst nahe an der "realen" Welt bleiben.
  - Anomalien vermeiden.
  - Nullwerte in Tupeln vermeiden.
  - Erzeugung von unechten Tupeln verhindern.

Semantische Integritätsbedingungen

- Ein DBMS bietet nicht nur Unterstützung bei der Speicherung und Verarbeitung von großen Datenmengen, sondern soll auch gewährleisten, dass eine Datenbank zu jedem Zeitpunkt ein korrektes Abbild der realen Welt widerspiegelt.
- Die zuvor behandelten Entwurfsrichtlinien unterstützen diese Anforderung, sind aber zu wenig formal, um einer objektiven Prüfung standzuhalten.
- Im Folgenden werden daher formale Richtlinien behandelt, die sogenannten **semantischen Integritätsbedingungen**.

Einschränkungen des Datenbankschemas

- **Einschränkungen des Wertebereichs dom (A)**
  Der Wertebereich einzelner Attribute wird genauer spezifiziert, als es die relativ allgemeine Angabe des Datentyps im Rahmen der Tabellendefinition ermöglich.
- **Festlegung von Schlüsseln und Eindeutigkeitsforderung**
  Unter den Schlüsselkandidaten (potenzielle Schlüssel) wird ein sog. Primärschlüssel definiert. Weiterhin kann festgelegt werden, ob sich alle Werte eines Attributes unterscheiden müssen.
- **Zulässigkeit von Nullwerten**
  Für Nichtschlüsselattribute kann festgelegt werden, ob Nullwerte zulässig sind.
- **Fremdschlüsselbeziehungen**
  Vereinbarungen zu sachlogischen Beziehungen zwischen Entities verschiedener Relationen.

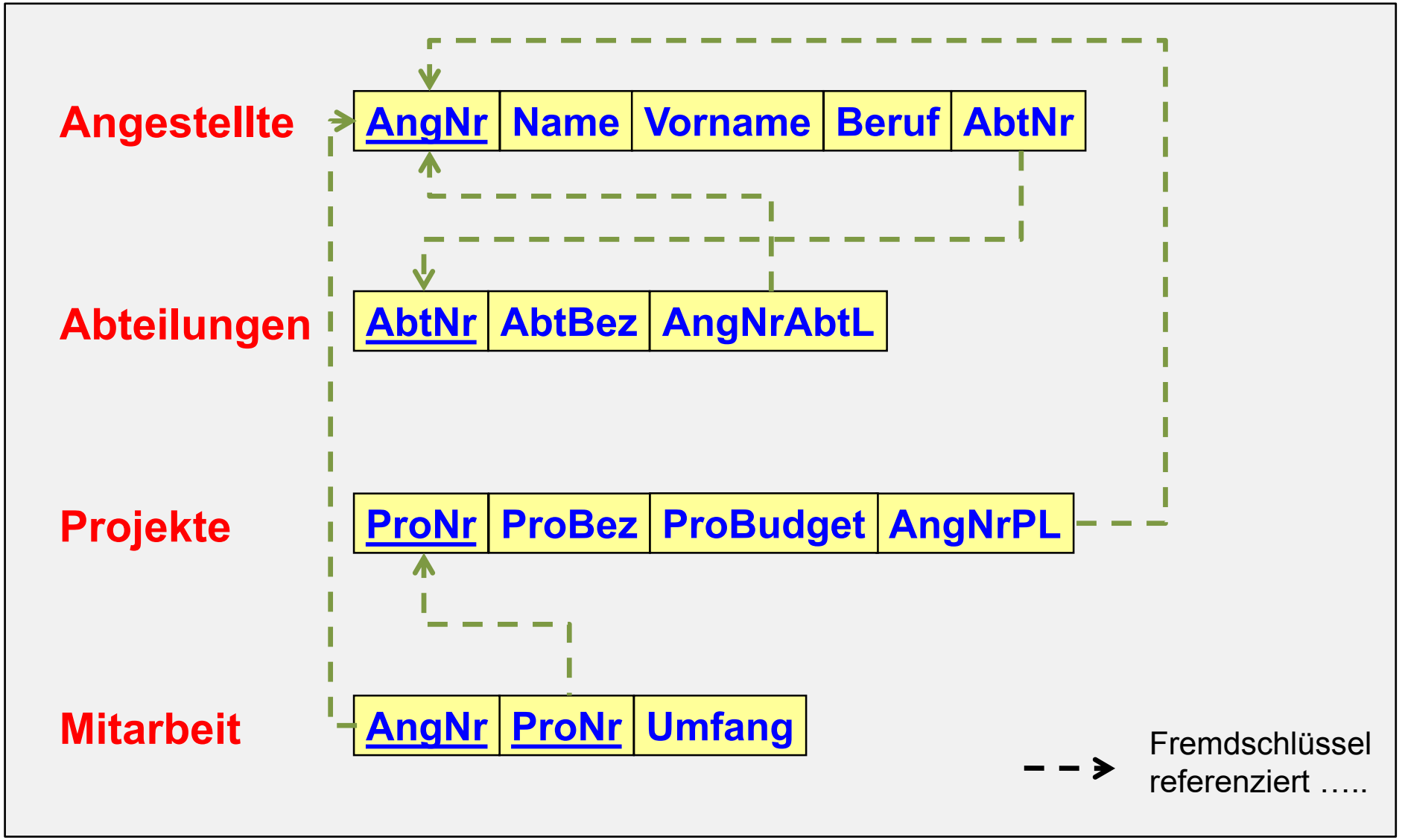
Semantische Integritätsbedingungen

- Gegeben sei ein Relationenschema R(A) mit
  - R = Relationenname,
  - A = $\{a_1, a_2, \ldots, a_n\}$ = Menge von Attributen, wobei jedes Attribut a_i einen atomaren Wertebereich dom(a_i) hat und dem
  - Wertebereich von R = dom(A) = dom(a_1) x dom(a_2) x … x dom(a_n)
- Eine semantische Integritätsbedingung (sIB) Σ über der Attributmenge A macht eine Aussage über Teilmengen X $\in$ dom (A) in der Form
  - X $\in$ dom (A) erfüllt Σ oder
  - X $\in$ dom (A) erfüllt Σ nicht.
- Fazit: Das Relationenschema wird ergänzt: R (A | Σ)

Interrelationale Semantische Integritätsbedingungen

- Analog kann man semantische Integritätsbedingungen zwischen Relationen definieren.
- Typisches Beispiel hierfür sind alle Aussagen bzw. Bedingungen bezüglich der Fremdschlüssel, die auch als **referenzielle Integritätsbedingungen** bezeichnet werden.
- Zur Erinnerung: Der Fremdschlüssel bezieht sich auf den Primärschlüssel oder eine bzw. mehrere UNIQUE-Spalten in der **referenzierten Tabelle**. Die Tabelle, in der der Fremdschlüssel definiert ist, bezeichnet man als **referenzierende Tabelle**.
- Beim Einfügen, Ändern und Löschen muss bzw. sollte festgelegt werden, welche Aktionen durchzuführen sind.

Relationales DB-Schema Firma (vgl. Elm-02, Seite 502 ff.)



Anwendungsbeispiel Fremdschlüssel-Bedingung

- Bezogen auf das einleitende Beispiel mit der Firma können beispielsweise folgende Bedingungen formuliert werden:
  - Angestellte.AbtNr $\subseteq$ Abteilung.AbtNr
  - Mitarbeit.ProNr $\subseteq$ Projekt.ProNr
  - Mitarbeit.AngNr $\subseteq$ Projekt.AngNr
- Durch eine zentrale und automatische Überprüfung der Integritätsbedingungen kann darauf verzichtet werden, derartige Mechanismen mehrfach zu implementieren.

Wiederholung: Folie aus Kapitel 4.2

- Tabellenbedingung::=
  {PRIMARY KEY | UNIQUE} (Spaltenname [, …])
  FOREIGN KEY (Spaltenname [, …])
  REFERENCES Tabellenname (Spaltenname [, …])
  [,ON DELETE {RESTRICT | NO ACTION | CASCADE | SET NULL | SET DEFAULT}]
  [,ON UPDATE {RESTRICT | NO ACTION | CASCADE | SET NULL | SET DEFAULT}]

- Primary key und unique gelten analog der Definition für Spaltenbedingungen, aber eben für mehrere Attribute.

- Ausführliche Erläuterungen zum Foreign Key und den damit verbunden Integritätsbedingungen folgen später noch.

Beispiele aus unserer Rechnerübung

```
create table Vorlesungen
    (VorlNr       integer primary key,
     Titel        varchar (40) not null,
     SWS          integer check (SWS between 1 and 9),
     gelesenVon   integer references Professoren on delete set null);

create table hören
    (MatrNr       integer references Studenten on delete cascade,
     VorlNr       integer references Vorlesungen on delete cascade,
     primary key (MatrNr, VorlNr));

create table voraussetzen
    (Vorgänger    integer references Vorlesungen on delete cascade,
     Nachfolger   integer references Vorlesungen on delete cascade,
     primary key (Vorgänger, Nachfolger));
```

- Ohne eine Überprüfung der referenziellen Integrität kann man leicht einen inkonsistenten Datenbankzustand erzeugen, beispielsweise durch
  - `Insert into Vorlesungen values (2000, ,Mathematik für Naturwissenschaftler', 4, 2222);`

Referenzielle Integritätsbedingungen in SQL

ON DELETE	Die definierte Fehlerkorrekturoption wird ausgeführt, wenn der referenzierte Master-Datensatz gelöscht wird.
ON UPDATE	Die definierte Fehlerkorrekturoption wird ausgeführt, wenn der Wert des referenzierten Schlüsselattributs in der Master-Tabelle geändert wird.
CASCADE	Die Detaildatensätze, deren Fremdschlüssel den manipulierten Master-Datensatz referenzieren, werden ebenfalls gelöscht (kaskadierendes Löschen) bzw. deren Schlüsselwert wird entsprechend geändert (kaskadierendes Ändern).
SET DEFAULT	Die Fremdschlüsselspalten des Detaildatensatzes, die den manipulierten Master-Datensatz referenzieren, werden auf den für diese Spalte definierten DEFAULT-Wert gesetzt.
SET NULL	Die Fremdschlüsselspalten des Detail-Datensatzes, die den manipulierten Master-Datensatz referenzieren, werden auf NULL gesetzt.
NO ACTION	Auf den Integritätsfehler wird je nach Prüfungszeitpunkt mit dem Zurückrollen der gesamten Transaktion bzw. der fehlerhaften DML-Anweisung reagiert (Reaktion wie bei den anderen CONSTRAINTS auch).
RESTRICT	Diese Option ist analog zur NO ACTION-Funktionalität zu sehen, mit dem kleinen Unterschied, dass bei RESTRICT vorübergehend während der Ausführung der UPDATE- oder DELETE-Anweisung Verletzungen des Fremdschlüssels hingenommen werden, solange am Ende die Bedingungen wieder erfüllt sind.

Quelle:
Fae-07,
Seite 219

Funktionale Abhängigkeit - Definition

- Gegeben sei ein Relationenschema R mit der Attributmenge $U = \{a_1, a_2, \ldots, a_n\}$ sowie zwei Teilmengen $A,B \subseteq U$.

  Anmerkung: In der Literatur wird im Rahmen der Relationentheorie die Arttibutmenge mit U (von „universal relation“) statt (wie bisher) mit A (für Attributmenge) bezeichnet. Die Vorlesung folgt dieser Nomenklatur.

- Eine semantische Integritätsbedingung heißt funktionale Abhängigkeit (fA) der Attributmenge B von der Attributmenge A wenn für alle $x,y \in X$ mit $X \in \text{dom}(U)$ und $A,B \subseteq U$ gilt:

$$\mathbf{x.A = y.A \Rightarrow x.B = y.B}$$

  Man schreibt dann auch A → B und sagt **B sei funktional abhängig von A**.

Funktionale Abhängigkeit – Praxisbeispiel

- Anschaulicher kann man die Definition wie folgt formulieren:
  - Wenn zwei Tupel x und y aus X in allen A-Werten identisch sind, dann stimmen sie auch in allen B-Werten vollständig überein.
  - In einer Tabelle folgt aus der Gleichheit der Zeilen in allen A-Spalten zwingend auch die Gleichheit der Zeilen in allen B-Spalten.
- Einfaches Beispiel für die Relation Angestellte:

AngNr	Name	Vorname	Beruf	AbtBez
4711	Weber	Claudia	Informatikerin	SWE
4712	Weber	Sabine	DV-Kauffrau	Technik
4713	Glück	Hans	BA-Student	SWE

  - Es gilt z.B. AngNr → Name und auch AngNr → AbtBez, nicht aber Name → AbtBez. In diesem Fall schreibt man Name $\not\rightarrow$ AbtBez

Schlüssel als Spezialfall funktionaler Abhängigkeiten

- Betrachtet man beispielsweise die folgende funktionale Abhängigkeit in unserer Relation Studenten
  MatrNr → Name, Vorname, Straße, PLZ, Ort, Telefon, Semester
  und ergänzt die rechte Seite um MatrNr (was möglich ist, da MatrNr → MatrNr offensichtlich, d.h. trivial, ist).

- Es fällt auf, dass auf der rechten Seite die komplette Relation steht. Folglich ist das gesamte Relationenschema von Studenten funktional abhängig von dem Attribut MatrNr.

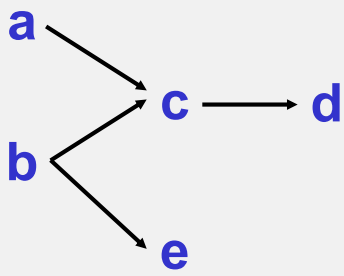
- Ein Schlüssel x liegt also genau dann vor, wenn für eine Relation R mit Attributmenge U und x $\in$ **U** gilt
  - x → U (U ist funktional abhängig von x) und
  - x ist minimal (es gibt keine echte Teilmenge von x mit derselben Eigenschaft)

Abhängigkeitsgraph – Definition und Beispiel

- Gegeben sei ein Relationenschema U sowie zwei Teilmengen A,B von U.
- Dann ist **F** definiert als eine Menge von gültigen funktionalen Abhängigkeiten und man schreibt

  $F(U) = \{A \rightarrow B \mid A,B \subseteq U\}$
- Die Menge F kann auch graphisch dargestellt werden:

  U = (a,b,c,d,e) und
  F= {ab→c, b→e,c→d}



- Aufgabe: Zeichnen Sie den Abhängigkeitsgraph für
  U = (a,b,c,d,e,f,g,h,i,j) und F= {a → b, cde → f, g → hij, j → a}

Spezielle funktionale Abhängigkeiten (1)

Triviale funktionale Abhängigkeiten

- Mit Regel 1 (Reflexivität) lassen sich auch funktionale Abhängigkeiten erzeugen, die immer wahr sind. Diese werden auch als **trivial** bezeichnet:
  Gegeben: R(U|F); $A, B \subseteq U$
  Eine funktionale Abhängigkeit $A \rightarrow B$ heißt trivial, wenn $B \subseteq A$
  - wegen $\varnothing \subseteq A \subseteq U$ gilt: **$A \rightarrow \varnothing$ und $U \rightarrow A$ sind trivial**
  - wegen $A \subseteq A$ gilt: **$A \rightarrow A$ ist trivial**

Einfache funktionale Abhängigkeiten

- Funktionale Abhängigkeiten, die auf der rechten Seite nur ein einzelnes Attribut enthalten, werden als auch **einfache funktionale Abhängigkeiten** bezeichnet.
  - Beispiel: Die funktionale Abhängigkeit $a \rightarrow bc$ kann (gemäß Dekompositionsregel) zerlegt werden in $a \rightarrow b$ und $a \rightarrow c$

Spezielle funktionale Abhängigkeiten (2)

- **Vollfunktionale und partielle funktionale Abhängigkeiten**
  - Gemäß der Erweiterungsregel können kann die linke Seite einer funktionalen Abhängigkeit beliebig mit Attributen aus U erweitert werden; man erhält dadurch weitere gültige funktionale Abhängigkeiten.
  - Eine funktionale Abhängigkeit wird als vollfunktional bezeichnet, wenn sie auf der linken Seite keine überflüssigen Attribute hat. Andernfalls bezeichnet man sie als partielle funktionale Abhängigkeit.

- Formal: Gegeben sei $R(U \mid F)$ und $A, B \subseteq U$ sowie eine nicht triviale funktionale Abhängigkeit $A \rightarrow B$
  - B ist **voll funktional** abhängig (von A), wenn es keine echte Teilmenge $A' \subset A$ gibt mit $A' \rightarrow B$ und man schreibt $A \overset{\bullet}{\rightarrow} B$
  - B ist **partiell** von A abhängig, wenn es eine funktionale Abhängigkeit gibt mit $A' \rightarrow B$ und $A' \subset A$.

partiell (eine/mehrere Sache/n streichbar ohne Aussage kaputt zu machen):
MatrNr -> Name
MatrNr, PLZ -> Name
MatrNr, GebDat -> Name
streichbar links PLZ und GebDat

vollfunktional (nichts kann gestrichen werden ohne Aussage kaputt zu machen):
MatrNr, VorlNr -> Note

Mögliche Klausur Aufgabe:
- Ist das partiell funktional- oder vollfunktional-abhängig)?
R(UIF)
U = {a,b,c,d,e}
F = {a->b, b->c, ac->b,de->a}
=> partiell funktional abhängig, da a->b und ac->b beide die selbe Aussage enthalten (mit a->b)

- M = {MatrNr, VorlNr, Note, Titel} (MatrNr, VorlNr sind primary)
F = {MatrNr,VorlNr -> Note; MatrNr,VorlNr -> Titel; MatrNr,VorlNr -> Note, Titel;
VorlNr -> Titel} => nur 2. Regel nur partiell abhängig von Primarys, wegen 4. Regel => Regeln 2. Normalform verletzt, 1. Regel ist vollfunktional abhängig => schlechte Abhängigkeit

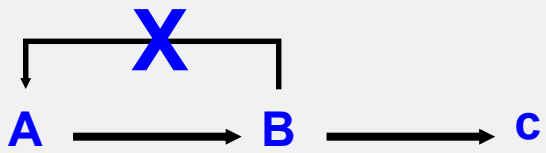
Spezielle funktionale Abhängigkeiten (3)

- Gegeben Sei eine Relation R mit der Attributmenge U sowie zwei Teilmengen A und B von U und ein Attribut c

  $R (U \mid F) \quad A,B \subseteq U, c \in U$

- Das Attribut c ist **transitiv abhängig** von A über B, wenn die folgenden Bedingungen erfüllt sind:

  1. $A \rightarrow B$
  2. $B \not\rightarrow A$
  3. $B \rightarrow c$
  4. $c \notin A \cup B$



- Anmerkungen:
  - (2) verhindert, dass A = B (im Sinne von austauschbar, weil sonst A → c gelten würde)
  - (4) verhindert, dass c trivialerweise (wegen Reflexivität) von A oder B abhängt.
  - Schreibweise: $A \overset{+}{\rightarrow} c$ bedeutet **c ist transitiv funktional Abhängig von A**

Inferenzregeln für funktionale Abhängigkeiten (1)

- Es gibt häufig eine Menge von funktionalen Abhängigkeiten, die semantisch offensichtlich sind, z.B. AngNr → AngName.
- Darüber hinaus existieren aber auch andere funktionale Abhängigkeiten, die auf den ersten Blick weniger offensichtlich sind, aber ebenfalls Gültigkeit besitzen, z.B. AngNr → PLZ.
- Um die systematische Herleitung aller gültigen funktionalen Abhängigkeiten zu erleichtern, gibt es die sogenannten Inferenzregeln.
- An dieser Stelle wird auf formale Beweise verzichtet; Interessierte mögen bitte ELM-02, Seite 513 ff. zu Rate ziehen.

Inferenzregeln für funktionale Abhängigkeiten (2)

Seien A,B,C,D Teilmengen der Attributmenge U einer Relation R

- **Reflexivität**
  Falls B eine Teilmenge von A ist, dann gilt immer auch A → B. Insbesondere gilt auch A → A
- **Erweiterungsregel**
  Falls A → B gilt, dann gilt auch AC → BC (bzw. auch AC → B)
- **Transitivität**
  Falls A → B und B → C gilt, dann gilt auch A → C
- **Vereinigungsregel**
  Falls A → B und A → C gelten, so gilt auch A → BC
- **Dekompositionsregel**
  Falls gilt A → B und C $\subseteq$ **B**, dann gilt auch A → C

Inferenzregeln für funktionale Abhängigkeiten (3)

- **Pseudotransitivitätsregel**
  Falls A → B und BC → D gelten, dann gilt auch AC → D

- Die ersten drei Regeln werden als Armstrong-Axiome (nach Armstrong,1974) bezeichnet. Sie sind korrekt und vollständig.
  - Die Korrektheit besagt, dass sich mit Hilfe der Armstrong-Axiome aus einer gegebenen Menge F von funktionalen Abhängigkeiten nur solche funktionalen Abhängigkeiten ableiten lassen, die von jeder Relationenausprägung erfüllt sind, für die F bereits erfüllt ist.
  - Die Vollständigkeit der Axiome besagt, dass sich hiermit alle gültigen funktionalen Abhängigkeiten ableiten lassen.

- Die restlichen drei Regeln wurden nur hinzugefügt, um den Herleitungsprozess etwas "komfortabler" zu gestalten.

Hüllenbildung (1)

- Betrachten wir folgende Beispielrelation (vgl. Kem-09, Seite 172 ff.):
  ProfAdressen <PersNr, Name, Vorname, Strasse, PLZ, Ort, BLand, Landesregierung, Einwohner, Vorwahl, Telefon, Rang, Raum>
  Annahmen: Orte seinen zumindest innerhalb der Bundesländer (=BLand) eindeutig, die PLZ ändere sich nicht innerhalb einer Straße, Straßen und Orte gehen nicht über Bundesland-grenzen hinweg und die Landesregierung stellt den jeweiligen Ministerpräsidenten eines BLandes (ist also funktional abhängig von diesem).

- Beim DB-Entwurf wurden folgende funktionale Abhängigkeiten erkannt:
  - PersNr → PersNr, Name, Vorname, Straße, PLZ, Ort, BLand, Landes-regierung, Einwohner, Vorwahl, Telefon, Rang, Raum
  - Ort, BLand → Einwohner, Vorwahl
  - PLZ → BLand, Ort, Einwohner
  - BLand → Landesregierung
  - Raum → PersNr
  - Ort, BLand, Straße → PLZ

Hüllenbildung (2)

- Es gibt jedoch noch weitere funktionale Abhängigkeiten, z.B.
  - PLZ → Landesregierung
  - Raum → PersNr, Name, Vorname, Strasse, PLZ, Ort, BLand, Landesregierung, Einwohner, Vorwahl, Telefon, Rang, Raum
- Gesucht bzw. von Interesse ist die Menge aller funktionalen Abhängigkeiten die aus einer gegebenen Menge F funktionaler Abhängigkeiten abgeleitet werden können. Diese wird auch als **F^+** bzw. **abgeschlossene Hülle von F** bezeichnet.
- Die Hülle F^+ einer Menge F von funktionalen Abhängigkeiten kann durch wiederholte Anwendung der Inferenzregeln bestimmt werden, was jedoch i.A. relativ aufwändig ist.

Verlustlosigkeit (siehe Kem.-09, S. 179-181)

- Zurück zum Praxisbeispiel aus Kap. 5.1

Lokal	Gast	Getränk
Bella Italia	Müller	Wein
Bella Italia	Schmitt	Bier
San Remo	Müller	Cola

- Warum hat es damals mit der Zerlegung nicht geklappt bzw. warum gilt :
  Getränkewahl <> *besucht* $\pi_{\text{Restaurant,Gast}}$ X *trinkt* $\pi_{\text{Gast,Getränk}}$
- Was genau sind die Kriterien, anhand derer man entscheiden kann, ob eine verlustfreie Zerlegung möglich ist?

Kriterien der Verlustlosigkeit (siehe Kem.-09, S.180)

- Die Zerlegung einer Relation R mit zugehörigen funktionalen Abhängigkeiten ist genau dann verlustlos, wenn mindestens eine der folgenden funktionalen Abhängigkeiten herleitbar ist:
  - $(R_1 \cap R_2) \rightarrow R_1 \in F^+_R$
  - $(R_1 \cap R_2) \rightarrow R_2 \in F^+_R$
- Im Weintrinkerbeispiel galt nur eine funktionale Abhängigkeit und zwar (Lokal, Gast) → Getränk.
- Es galt jedoch nicht eine der beiden Bedingungen für Verlust-losigkeit nämlich
  - (Lokal, Gast ∩ Gast, Getränk) = **Gast → Getränk** $\in F^+_R$
  - (Lokal, Gast ∩ Gast, Getränk) = **Gast → Lokal** $\in F^+_R$

Der Aplus-Algorithmus (1)

- Das Problem, die Hülle F+ zu einer Menge von funktionalen Abhängigkeiten zu bestimmen, ist nicht linear und verursacht entsprechend hohen Rechenaufwand.

- Oftmals ist man jedoch nicht die gesamte Hülle F^+ einer Menge von funktionalen Abhängigkeiten interessant, sondern nur die Menge von Attributen A^+, die von der Attributmenge A gemäß der Menge F funktional bestimmt wird.

- Formal: Gegeben sei R(U|F) und $A \subseteq U$.
  Für eine beliebige Menge A von Attributen erhält man bei gegebenem F unter Anwendung der Armstrong-Regeln die **Hülle A^+** von A , d.h. Menge aller von A funktional abhängigen Attribute
  - $\mathbf{A^+} = \{b \in U \mid A \rightarrow b\}$

Anwendungsbeispiel für APlus

- Gegeben sei die Relation R mit der Attributmenge A = {a,b,c,d,e} und die Menge F der funktionalen Abhängigkeiten mit
  F = {ab → c, b → ce, c → d, a → d}

- Bestimmen Sie b^+

A^+	F'
b	{ab → c, b → ce, c → d, a → d}
bce	{ab → c, c → d, a → d}
bcde	{ab → c, a → d}
bcde	

- Bestimmen Sie ab^+

A^+	F'
ab	{ab → c, b → ce, c → d, a → d}
abc	{b → ce, c → d, a → d}
abcd	{b → ce, a → d}
abcde	{a → d}
abcde	

ab ist Schlüssel, weil a^+ ebenfalls vorzeitig terminiert. ⇦

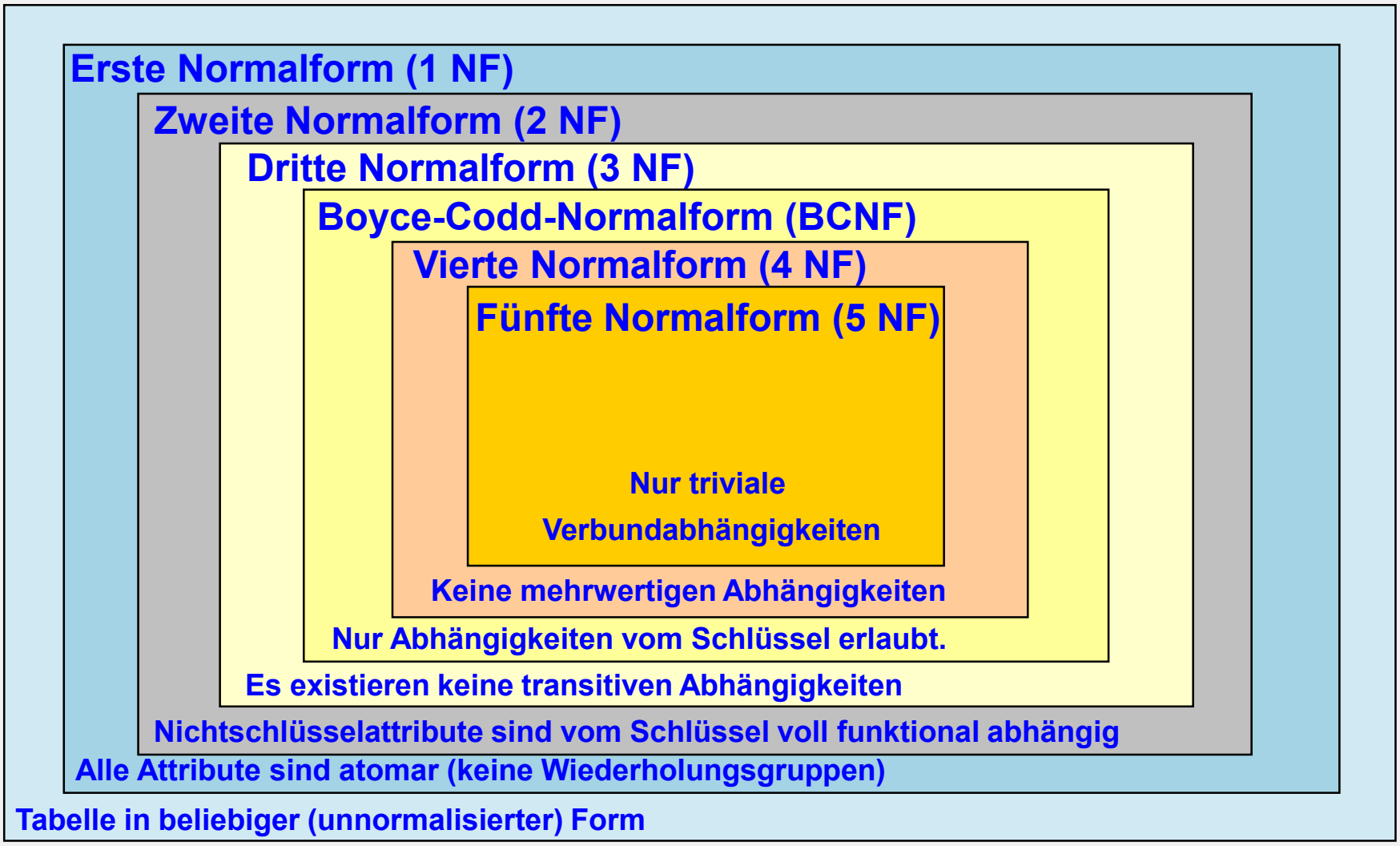
Wozu Normalformen?

- Die wissenschaftlichen Untersuchungen im Rahmen des Relationenmodells hat eine Datenbanktheorie hervorgebracht, bei der die formalen Aspekte – teilweise unabhängig von den praxisorientierten Gegebenheiten – präzise beschrieben werden.

- Ein bedeutendes Teilgebiet der Datenbanktheorie bilden die sogenannten Normalformen. Mit diesen werden innerhalb von Relationen Abhängigkeiten aufgezeigt, formalisiert und analysiert, um die einleitend dargestellte Redundanzen und die damit verbundenen Anomalien zu vermeiden.

- Die bisher in Kapitel 5 eingeführten Begriffe dienen allesamt dem besseren Verständnis der nachfolgend beschriebenen Normalformen. Basierend auf der Struktur funktionaler Abhängigkeiten sollen hierdurch Redundanzen vermieden und eine mehr formalisierte Vorgehensweise beim Aufbau von Relationenschemata erreicht werden.

Übersicht Normalformen



Quelle: Mei-01, S. 35

Erste Normalform (1NF)

Eine Relation ist in der ersten Normalform (1 NF), wenn alle Attribute atomare Wertebereiche (Domänen) besitzen.

- Die erste Normalform gilt heute bereits als Teil der formalen Definition einer Relation. Aus historischer Sicht wurde sie definiert, um mehrwertige und zusammengesetzte Attribute auszuschließen.
- 1 NF verhindert, dass eine Werte-Menge ein Attributwert für ein Tupel sein kann.
  M.a.W. verhindert 1NF Relationen innerhalb von Relationen.

Zweite Normalform (2NF)

Eine Relation R mit zugehörigen FDs (funktionalen Abhängigkeiten) F ist in der zweiten Normalform, wenn sie

- **in der ersten Normalform ist und**
- **jedes Nichtschlüssel-Attribut von jedem Schlüssel(kandidaten) voll funktional abhängig ist der Relation.**

- Eine Relation ist demnach nicht in der 2NF, wenn irgendein NSA von irgendeinem Schlüssel nur partiell abhängig ist.
- Intuitiv verletzt eine Relation die Forderung nach Erfüllung der 2NF meist dann, wenn in der der Relation Informationen über mehr als ein Konzept der realen Welt (Entity-Typ) modelliert werden.

Anschauliches Beispiel für 2NF (1)

- Betrachten wir die folgende Relation Prüfung :

MatrNr	VorlNr	Name	Note
54164	2009	Baumgärtner	1,7
54190	2009	Fath	2,3
54190	2010	Fath	1,9
52125	2017	Bergmann	1,4
52125	2022	Bergmann	2,9
52125	2025	Bergmann	2,1

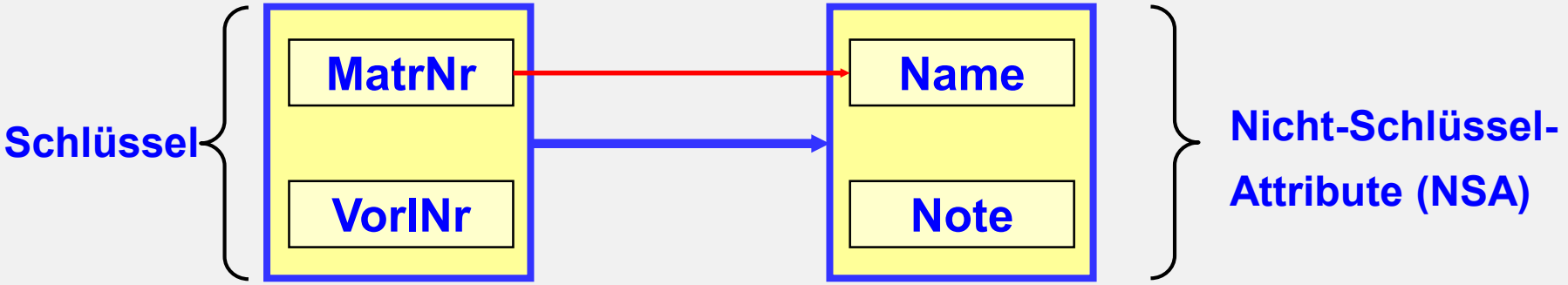
- Der Schlüssel ist <MatrNr, VorlNr> (falls bei einer Nachklausur das Ergebnis überschrieben wird) und es existiert die funktionale Abhängigkeit

  MatrNr, VorlNr → MatrNr, VorlNr, Name, Note.

- Allerdings existieren auch weitere funktionale Abhängigkeiten, z.B. MatrNr → Name und MatrNr, VorlNr → Note

Anschauliches Beispiel für 2NF (2)

- Die Situation lässt sich auch durch einen Abhängigkeitsgraphen darstellen:



- Da das Attribut Name als NSA bereits von einem Teil des Schlüssels, nämlich der MatrNr, voll funktional abhängig ist, erfüllt diese Relation nicht die Forderung der 2NF.

- Das Beispiel demonstriert nochmals die daraus resultierenden Probleme:
  - Einfügeanomalie: Was macht man mit Studenten, die (noch) keine Vorlesung hören?
  - Updateanomalie: Wenn z.B. Bergmann in das 9. Semester kommt, muss sichergestellt sein, dass alle Tuppel (im Beispiel 3 Einträge) geändert werden.
  - Löschanomalie: Was passiert, wenn Baumgärtner seine einzige Vorlesung absagt?

Dritte Normalform (3NF)

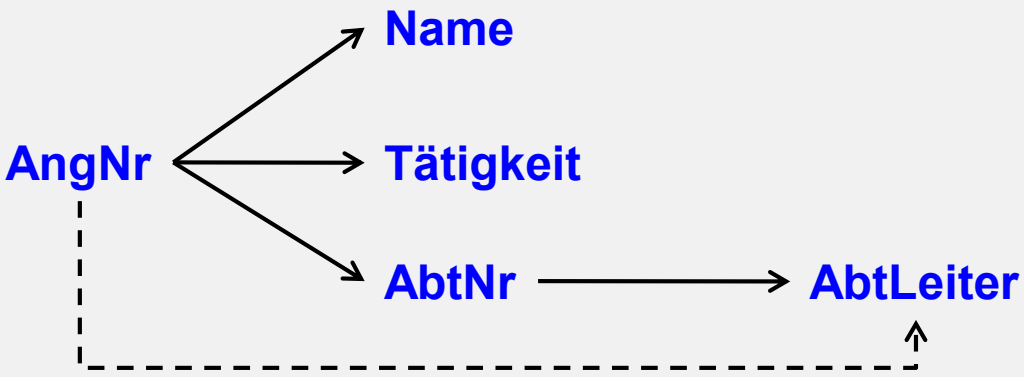
Eine 1NF Relation R ist in der dritten Normalform, wenn kein Nichtschlüsselattribut (NSA) von irgendeinem Schlüsselkandidaten transitiv abhängig ist.

- Ein Relationenschema ist demnach nicht in der dritten Normalform, wenn irgendein Nichtschlüsselattribut von irgendeinem Schlüsselkandidaten transitiv abhängig ist.
- Außerdem gilt für eine Relation R
  - R nicht in 2NF $\Rightarrow$ R auch nicht in 3NF

Anschauliches Beispiel für 3NF

- Gegeben sei das folgende Relationenschema R1(U | F) mit
  U = { AngNr, Name, Tätigkeit, AbtNr, AbtLeiter} und
  F = { AngNr → Name, Tätigkeit, AbtNr; AbtNr → AbtLeiter}

- Abhängigkeitsgraph:



- Frage: Ist diese Relation in 2NF und in 3NF? nicht in 3NF, sondern nur in 2NF

Abstraktes Beispiel für 3NF

- Gegeben sei R = (U | F) mit
  - U = <a,b,c,d,e>
  - F = {ab → c, c → d, b → e}
- Bestimmen Sie den Schlüssel und die Nichtschlüsselattribute.
- Befindet sich diese Relation in 3NF?

Probleme mit 3 NF (Bsp. in Anlehnung an Kem.-09, S. 189)

- Betrachten wir folgende Relation

  Städte (Ort, BLand, Ministerpräsident, Einwohner)

  mit folgenden funktionalen Abhängigkeiten

  - fA1: Ort, BLand → Einwohner
  - fA2: BLand → Ministerpräsident
  - fA3: Ministerpräsident → BLand

- Frage: In welcher Normalform ist diese Relation?

* Annahme, dass Orte innerhalb eines Bundeslandes eindeutig sind.

Probleme der 3NF / Def. Superschlüssel

- Die Beispielrelation ist zwar in 3NF; trotzdem tauchen Anomalien auf, z.B. wenn sich der Ministerpräsident eines Bundeslandes ändert.
- Dies liegt daran, dass bei 3NF auf transitive Abhängigkeit der NSA von den Schlüsselkandidaten geprüft wird, nicht aber Redundanzen innerhalb der Schlüsselattribute selbst. Das ist hier aber das Problem.
- **Superschlüssel**: In einer Relation R mit der Attributmenge U und einer Teilmenge $A \subseteq R$ bezeichnet man A als Superschlüssel von R wenn gilt $A \rightarrow U$.
- M. a. W. enthält A einen Schlüsselkandidaten, ist aber im Gegensatz zu diesem nicht minimal. Insbesondere gilt natürlich $U \rightarrow U$.

BCNF (Boyce-Codd-Normalform)

Eine 1NF Relation R mit FDs F ist in der BCNF, falls für jede nicht-triviale funktionale Abhängigkeit A → b gilt:

- **A ist Superschlüssel (d.h. A enthält einen Schlüssel)**

- Anmerkung:
  - Für jede elementare, nicht-triviale funktionale Abhängigkeit A → b gilt: **A ist Schlüssel**.
  - Folglich gehen bei BCNF alle elementaren, nicht-trivialen funktionalen Abhängigkeiten von Schlüsseln aus.
- Weiterhin gilt:
  - R nicht in 2NF $\Rightarrow$ R nicht in 3NF $\Rightarrow$ R nicht in BCNF
  - R nicht in 3NF $\Rightarrow$ R nicht in BCNF

Dekompositionsalgorithmus (1)

- Problemstellung: Gesucht ist ein Algorithmus, mit dem man aus einer Relation, welche die BCNF nicht erfüllt, mehrere (kleinere) Relationen erzeugen kann, die die BCNF erfüllen.
- Gleichzeitig muss diese Zerlegung „verlustlos“ sein (siehe Gegenbeispiel in Abschnitt 5.1 mit dem Weintrinker)
- Formales Ziel: $Z = \{R_1, R_2, \ldots, R_n\}$ sei eine Menge von verlustlosen Zerlegungen der Relation R, die alle BCNF erfüllen.

Dekompositionsalgorithmus (2)

- Starte mit Z = {R}
- Solange es noch ein Relationenschema $R_i \in Z$ gibt, das nicht die Forderung BCNF erfüllt, mache folgendes:

1. **Finde** eine für R_i geltende, nicht triviale funktionale Abhängigkeit

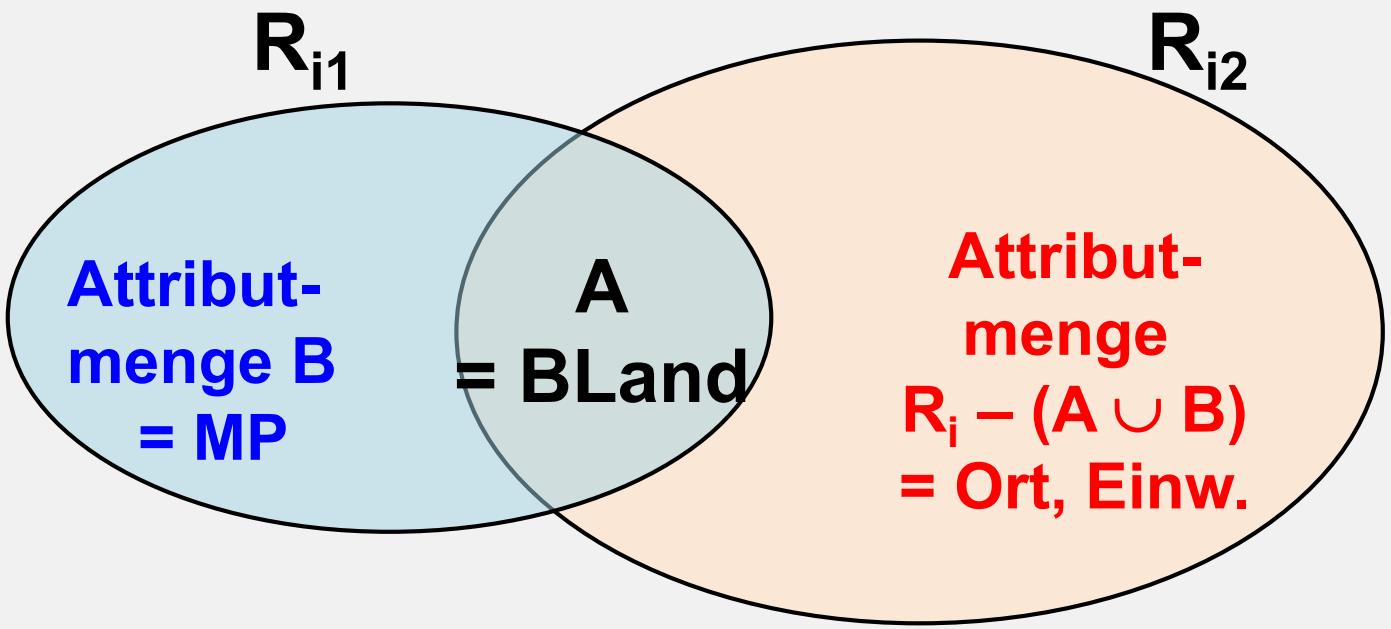
   $A \rightarrow B$ mit $A \cap B = \varnothing$ und $A \rightarrow R_i$

   Hinweis: Man sollte die funktionale Abhängigkeit so wählen, dass B alle von A funktional abhängigen Attribute enthält – also $B \in (R_i - A)$.

2. **Zerlege** R_i in die beiden Schemata $R_{i1} := A \cup B$ und $R_{i2} := R_i - B$

3. **Entferne** R_i aus Z und füge R_{i1} und R_{i2} in Z ein.

Grafik, die die Zerlegung abstrakt illustriert



Entnommen aus Kem.09, Seite 190

Mehrwertige Abhängigkeiten (1)

- Basis des klassischen Relationenmodells von Codd ist die Forderung, dass Attribute grundsätzlich atomar, d.h. nicht mehrwertig, sein müssen. Dies impliziert auch die erste Normalform.

- Beispiel: Sprachkenntnis eines Angestellten und alternative Abbildung in einer Relation:

AngNr	Name	Sprachen
4711	Weber	{deutsch}
4712	Schmitt	{deutsch, englisch, spanisch}
4713	Maier	{deutsch, englisch}

1NF

AngNr	Name	Sprache
4711	Weber	deutsch
4712	Schmitt	deutsch
4712	Schmitt	englisch
4712	Schmitt	spanisch
4713	Maier	deutsch
4713	Maier	englisch

Mehrwertige Abhängigkeiten (2)

- Spätestens wenn ein weiteres mehrwertiges Attribut – also eine zweite Attributmenge abzubilden ist – muss das kartesische Produkt aller möglichen Kombinationen gebildet werden.

- Beispiel: Sprachkenntnis und Hobby eines Angestellten:

AngNr	Name	Sprachen	Hobbies
4711	Weber	{deutsch}	{wandern, lesen}
4712	Schmitt	{deutsch, englisch, spanisch}	{kochen, lesen, singen}
4713	Maier	{deutsch, spanisch}	{segeln, lesen}

1NF

Hier treten Anomalien auf – welche z.B.?

Definition mehrwertige Abhängigkeit (1)

- Gegeben sei R(U|F‘) mit A,B $\subseteq$ U und F‘ eine Menge von funktionalen und mehrwertigen Abhängigkeiten, gültig für U

- **B ist mehrwertig abhängig von A (Schreibweise A →→ B) ⇔ für alle Tupel x, y aus r gilt: falls x.A = y.A ⇒ es existieren Tupel u, v mit**
  - **u.AB = x.AB,**
  - **u.(U\AB) = y. (U\AB),**
  - **v.AB=y.AB,**
  - **v.(U\AB)=x.(U\AB)**

Definition mehrwertige Abhängigkeit (2)

- Andere, möglichweise einprägsamere Formulierung:
- Gegeben sei a ∈ dom (A), b ∈ dom (B), c ∈ dom (C)
- Falls es in R Tupel x,y gibt mit
  - x=(a b c)
  - y=(a b' c')

  so muss es Tupel u,v geben mit
  - u=(a b c')
  - v=(a b' c)

Vierte Normalform (4NF)

Eine 1NF Relation R mit funktionalen und mehrwertigen FDs F‘ ist in der vierten Normalform (4NF), falls für jede nicht-triviale mehrwertige Abhängigkeit A →→ B gilt:

- **A ist Superschlüssel (d.h. A enthält einen Schlüssel)**

- M.a.W. wird bei der 4NF die durch mehrwertige Abhängigkeiten verursachte Redundanz ausgeschlossen.
- Weiterhin gilt:
  - R nicht in 2NF $\Rightarrow$ R nicht in 3NF $\Rightarrow$ R nicht in BCNF $\Rightarrow$ R nicht in 4NF bzw.
  - R in 4NF $\Rightarrow$ R in BCNF $\Rightarrow$ R in 3NF $\Rightarrow$ R in 2NF

Verlustlose Zerlegung

- Ein Relationenschema R (U|F‘) mit A,B $\subseteq$ U kann genau dann verlustlos in die beiden Schemata R_A und R_B zerlegt werden, wenn gilt:
  - $R = R_A \cup R_B$ und

  mindestens es existiert eine der beiden mehrwertigen Abhängigkeiten
  - $R_A \cap R_B \rightarrow R_A$ oder
  - $R_A \cap R_B \rightarrow R_B$
- Beispiel: Die Relation Angestellte (AngNr, Sprache, Hobby) kann (verlustlos!) zerlegt werden in
  R_A = (AngNr, Sprache) und
  R_B = (AngNr, Hobby)

Abschließendes Beispiel (vgl. Kem-09, S. 194)

- Betrachten wir die Relation aus der Universitäts-Datenbank
  - Assistenten (PerNr, Name, Fachgebiet, Boss, Sprache, Programmier-sprache)

  mit der funktionalen Abhängigkeit
  - PersNr → Name, Fachgebiet, Boss

  sowie den beiden mehrwertigen Abhängigkeiten
  - PersNr →> Sprache
  - PersNr →> Programmiersprache

- Diese Relation ist mit Sicherheit nicht in der 4NF; sie ist nicht einmal in der 2NF (warum?). Bitte führen Sie eine geeignete Zerlegung durch, so dass alle Relationen 4NF erfüllen.