

Datenbanken

Relationale Entwurfstheorie

Nikolaus Augsten

nikolaus.augsten@sbg.ac.at

FB Computerwissenschaften
Universität Salzburg

Wintersemester 2013/14

Themenübersicht – Relationale Entwurfstheorie

- Richtlinien für den relationalen Entwurf
- Funktionale Abhängigkeiten
- Zerlegungen: Verlustlosigkeit und Abhängigkeitsbewahrung
- Normalformen: 1NF, 2NF, 3NF, BCNF

Inhalt

- 1 Richtlinien für Relationalen Datenbankentwurf
- 2 Funktionale Abhängigkeiten
- 3 Zerlegung von Relationen
- 4 Normalformen

Richtlinien für Relationalen Datenbankentwurf

Inhalt

- 1 Richtlinien für Relationalen Datenbankentwurf
- 2 Funktionale Abhängigkeiten
- 3 Zerlegung von Relationen
- 4 Normalformen

Ziele des Relationalen Datenbankentwurfs

- Ziel des relationalen Entwurfs ist ein Satz guter relationaler Schemas.
- Die Hauptschwierigkeit ist es, eine gute Gruppierung der Attribute in relationale Schemas zu finden.
- Gute relationale Schemas zeichnen sich folgendermaßen aus:
 - einfache und klare Semantik (Bedeutung) von Tupeln und Attributen
 - Vermeidung von redundanten Daten
 - Vermeidung von Anomalien bei Datenänderungen
 - Vermeidung von *null* Werten soweit möglich
 - Nur genau die ursprünglichen Daten sind gespeichert und natürliche Joins erzeugen keine zusätzlichen Tupel

Anomalien bei Datenänderung/1

- Beispiel Schema:
 - AngProj(SVN, PNum, Stunden, AName, PName, POrt)
- Updateanomalie
 - Wenn der Ort eines Projektes geändert wird, muss er für alle Angestellten im Projekt geändert werden.
- Einfügeanomalie
 - Es kann kein Projekt ohne Angestellte eingefügt werden (außer mithilfe von *null* Werten).
- Löschanomalie
 - Wenn ein Projekt gelöscht wird, werden als Nebeneffekt auch alle Angestellten gelöscht, die auf diesem Projekt arbeiten.

Anomalien bei Datenänderung/2

- Beispiel Instanz zum Schema:
AngProj(SVN, PNum, Stunden, AName, PName, Ploc)
- | AngProj | | | | | |
|---------|------|---------|---------|-----------------|-----------|
| SVN | PNum | Stunden | AName | PName | POrt |
| 1234 | 1 | 32.5 | Schmidt | ProductX | Salzburg |
| 1234 | 2 | 7.5 | Schmidt | ProductY | Wien |
| 6688 | 3 | 40.5 | Mair | ProductZ | Linz |
| 4567 | 1 | 20.0 | Huber | ProductX | Salzburg |
| 4567 | 2 | 20.0 | Huber | ProductY | Wien |
| 3334 | 2 | 10.0 | Wong | ProductY | Wien |
| 3334 | 3 | 10.0 | Wong | ProductZ | Linz |
| 3334 | 10 | 10.0 | Wong | Computerization | Innsbruck |
| 3334 | 20 | 10.0 | Wong | Reorganization | Linz |
- AngProj ist kein gutes relationales Schema, da es unter Anomalien leidet.

Richtlinien für den relationalen Entwurf

- **Richtlinie 1:** Jedes Tupel einer Relation sollte nur die Instanz *einer* Entität oder Beziehung darstellen.
- **Richtlinie 2:** Update-, Einfüge- und Löschanomalien sollen vermieden werden.
- **Richtlinie 3:** Die Relationen sollen möglichst wenige *null* Werte enthalten; Attribute mit *null* Werten kommen in eine eigene Relation (zusammen mit dem Primärschlüssel).
- **Richtlinie 4:** Durch einen natürlichen Join von Relationen sollen keine zusätzlichen (d.h. falschen) Tupel erzeugt werden.

Inhalt

- 1 Richtlinien für Relationalen Datenbankentwurf
- 2 Funktionale Abhängigkeiten
- 3 Zerlegung von Relationen
- 4 Normalformen

Schlüssel (Auffrischung)

- Ein **Superschlüssel** einer Relation $R[A_1, A_2, \dots, A_n]$ ist eine Menge von Attributen $S \subseteq sch(R)$, sodass für keine zwei Tupel $t_1 \in R$ und $t_2 \in R$ einer beliebigen gültigen Ausprägung von R gilt: $t_1[S] = t_2[S]$.
- Ein **Kandidatschlüssel** K ist ein Superschlüssel für den gilt, dass durch die Entfernung eines beliebigen Attributes von K die Superschlüssel-Eigenschaft von K verloren geht.
- Eine *beliebiger* Kandidatenschlüssel wird als **Primärschlüssel** ausgewählt.
- *Notation*: Die Attribute des Primärschlüssels werden unterstrichen: AngProj(SVN, PNum, Stunden, AName, PName, Ploc)

- Definition
- Armstrong-Axiome
- Richtigkeit und Vollständigkeit
- Hülle und kanonische Überdeckung

Funktionale Abhängigkeiten/1

- **Funktionale Abhängigkeiten** (FDs – functional dependencies) werden zwischen Attributmengen $X \subseteq sch(R)$ und $Y \subseteq sch(R)$ einer Relation R definiert.
- **Definition**: Y ist von X **funktional abhängig** genau dann, wenn der Wert von X einen eindeutigen Wert von Y in R vorgibt:

$$X \rightarrow Y \Leftrightarrow \forall t_1, t_2 \in R : t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$$

- $X \rightarrow Y$ bedeutet, dass Y von X **funktional abhängt**, bzw., dass die Attribute X die Attribute Y **funktional bestimmen**.
- FDs definieren eine **Einschränkung auf das Schema**, d.h., auf alle möglichen relationalen Instanzen von R .
- **Definition**: Eine Menge Y ist **trivial funktional abhängig** von X genau dann wenn $Y \subseteq X$.

Funktionale Abhängigkeiten/2

- Funktionale Abhängigkeiten werden als **formales Maß** für die Qualität eines relationalen Entwurfs verwendet.
- Funktionale Abhängigkeiten und Schlüssel werden verwendet, um **Normalformen** für Relationen zu definieren.
- Funktionale Abhängigkeiten sind **Einschränkungen**, die abgeleitet werden von
 - der Bedeutung der Attribute,
 - der Beziehung der Attribute untereinander.
- Funktionale Abhängigkeiten werden von Einschränkungen der **zugrundeliegende Anwendung** auf die Attribute abgeleitet.
- **Notation:**
 - Statt $\{A, B\}$ schreiben wir AB (oder A, B), z.B. $AB \rightarrow BCD$ (statt $\{A, B\} \rightarrow \{B, C, D\}$).
 - Für eine Menge von Attributen X (z.B., $X = \{A, B, C\}$) und ein einzelnes Attribut A schreiben wir $X - A$ statt $X - \{A\}$.

Integrierte Übung 5.1

Betrachten Sie die abgebildete Instanz der Relation $R[A, B, C]$. Welche der folgenden Aussagen sind korrekt?

R		
A	B	C
1	1	3
2	1	1
3	2	2
4	1	1

- $B \rightarrow C$ gilt für die Relation R .
- $C \rightarrow B$ gilt für die Relation R .
- $BC \rightarrow A$ gilt in der abgebildeten Instanz von R .
- A ist der Primärschlüssel von R .

Funktionale Abhängigkeiten/3

Beispiele Funktionaler Abhängigkeiten:

- Sozialversicherungsnummer bestimmt Angestelltenname:
 - $SVN \rightarrow AName$
- Projektnummer bestimmt Projektname und Projektort:
 - $PNum \rightarrow PName, POrt$
- Angestellten SVN und Projektnummer bestimmten die Anzahl der Wochenstunden, die der Angestellte auf dem Projekt arbeitet:
 - $SVN, PNum \rightarrow Stunden$

Funktionale Abhängigkeiten/4

- FDs sind auf dem Schema definiert und müssen für alle Instanzen gelten.
- Gewisse FDs können aufgrund einer gegebenen Instanz der Datenbank ausgeschlossen werden:

Lehre

Professor	Kurs	Lehrbuch
Schmidt	Data Structures	Bertram
Schmidt	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz

Lehrbuch \rightarrow Kurs ist eine mögliche FD

Professor \rightarrow Kurs ist nicht erfüllt

FDs und Schlüssel

- Falls $K \subseteq sch(R)$ ein **Superschlüssel** von R ist, dann gilt

$$K \rightarrow sch(R),$$

d.h. K bestimmt *alle* Attribute in R .

- Falls $K \subseteq sch(R)$ ein **Kandidatschlüssel** von R ist, dann muss gelten:

- $K \rightarrow sch(R)$
- K kann nicht mehr verkleinert werden, ohne die Superschlüssel-Eigenschaft zu verlieren, d.h.,

$$\forall A \in K : (K - A) \not\rightarrow sch(R)$$

Armstrong-Axiome/1

- Für eine bestimmte Menge F von FDs können zusätzliche FDs hergeleitet werden, die immer gelten, wenn die FDs in F gelten.
- Armstrong-Axiome¹** (Inferenzregeln):
 - Reflexivität: $Y \subseteq X \models X \rightarrow Y$
 - Verstärkung: $X \rightarrow Y \models XZ \rightarrow YZ$
 - Transitivität: $X \rightarrow Y, Y \rightarrow Z \models X \rightarrow Z$
- Notation:**
 - $A \models B$ heißt: von A kann B hergeleitet werden
 - XZ steht für $X \cup Z$
- Die Armstrong-Axiome sind **korrekt** und **vollständig**:
 - Diese Regeln sind gültig (korrekt) und alle anderen gültigen Regeln können von diesen Regeln abgeleitet werden (vollständig).

¹William W. Armstrong: Dependency Structures of Data Base Relationships, pages 580-583. IFIP Congress, 1974.

Integrierte Übung 5.2

Zeige oder widerlege folgende Herleitungen:

- $W \rightarrow Y, X \rightarrow Z \models WX \rightarrow Y$
- $X \rightarrow Y, Z \subseteq Y \models X \rightarrow Z$
- $X \rightarrow Y, X \rightarrow W, WY \rightarrow Z \models X \rightarrow Z$
- $XY \rightarrow Z, Y \rightarrow W \models XW \rightarrow Z$
- $X \rightarrow Z, Y \rightarrow W \models X \rightarrow Y$
- $X \rightarrow Y, XY \rightarrow Z \models X \rightarrow Z$

Armstrong-Axiome/2

- Folgende **zusätzliche Inferenzregeln** werden oft verwendet:
 - Dekompositionsregel: $X \rightarrow YZ \models X \rightarrow Y, X \rightarrow Z$
 - Vereinigungsregel: $X \rightarrow Y, X \rightarrow Z \models X \rightarrow YZ$
 - Pseudotransitivitätsregeln: $X \rightarrow Y, WY \rightarrow Z \models WX \rightarrow Z$
- Diese zusätzlichen Inferenzregeln (und alle anderen möglichen Inferenzregeln) lassen sich aufgrund der Vollständigkeit der Armstrong-Axiome aus diesen ableiten.

Hülle/1

- Die Hülle F^+ (closure) der Menge F von FDs ist die Menge aller FDs die von F hergeleitet werden können.
- Die Hülle X^+ einer Menge von Attributen X bezüglich F ist die Menge aller Attribute, welche aus X hergeleitet werden können.
- Notation: $\mathcal{AH}(F, X)$ ist die Hülle X^+ bezüglich F .
- F^+ und X^+ können durch wiederholte Anwendung der Armstrong-Axiome berechnet werden.

Hülle/2

- Die Attribut-Hülle $X^+ = \mathcal{AH}(F, X)$ der Attributmenge X bezüglich F kann auch durch folgenden Algorithmus berechnet werden.

```

 $\mathcal{AH}(F, X)$ 
   $Erg := X$ 
  while (Änderungen an  $Erg$ ) do
    foreach FD  $A \rightarrow B$  in  $F$  do
      if  $A \subseteq Erg$  then  $Erg := Erg \cup B$ 
  return  $Erg$ 

```

- Input: Menge F von FDs, Attributmenge X
- Output: Attributhülle X^+ bezüglich F

Überdeckung und Äquivalenz

- F ist eine Überdeckung von G (F covers G) wenn jede FD in G von F hergeleitet werden kann, i.e., $G^+ \subseteq F^+$.
- Zwei Mengen F und G von FDs sind äquivalent genau dann wenn $F^+ = G^+$.
- Gleichbedeutende Definitionen von äquivalent:
 - Jede FD in F kann von G hergeleitet werden und jede FD in G kann von F hergeleitet werden.
 - F ist eine Überdeckung von G und G ist eine Überdeckung von F .

Membership und Äquivalenz

- Membership-Problem: Ist $X \rightarrow Y$ in F^+ ?
- Das Membership-Problem für $X \rightarrow Y$ und einer Menge F von FDs kann folgendermaßen ausgedrückt werden:

$$X \rightarrow Y \in F^+ \Leftrightarrow Y \subseteq \mathcal{AH}(F, X)$$

- Der Algorithmus zur Berechnung der Attributhülle kann also für das Membership-Problem angewandt werden.
- Membership-Algorithmus zur Äquivalenz zwischen F und G :
 1. Teste für alle FDs in F ob sie in G^+ sind.
 2. Teste für alle FDs in G ob sie in F^+ sind.
 3. F und G sind genau dann äquivalent, wenn alle Membership-Tests erfolgreich waren.

Integrierte Übung 5.3

Betrachte $F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H\}$ und $G = \{A \rightarrow CD, E \rightarrow AH\}$. Sind F und G äquivalent?

Zeigen oder widerlegen Sie die Äquivalenz auf zwei Arten:

1. Armstrong-Axiome
2. Membership-Algorithmus

Kanonische Überdeckung

- Zu einer gegebenen Menge F von FDs nennt man F_c eine **kanonische Überdeckung** wenn folgende drei Eigenschaften erfüllt sind:
 1. $F_c^+ = F^+$ (d.h., F_c und F sind äquivalent)
 2. In F_c existieren keine FDs $X \rightarrow Y$, bei denen X oder Y überflüssige Attribute enthalten, d.h., es muss gelten:
 - Keine FD $X \rightarrow Y$ in F_c kann durch $X' \rightarrow Y$ mit $X' \subseteq X$ ersetzt werden ohne die Äquivalenz zu F zu verletzen.
 - Keine FD $X \rightarrow Y$ in F_c kann durch $X \rightarrow Y'$ mit $Y' \subseteq Y$ ersetzt werden ohne die Äquivalenz zu F zu verletzen.
 3. Jede linke Seite einer funktionalen Abhängigkeit in F_c ist einzigartig.
- Die kanonische Überdeckung ist sozusagen eine **minimale Menge von FDs** welche noch äquivalent ist zu F .
- Jedes Menge F von FDs hat eine kanonische Überdeckung.
- Es kann mehrere kanonische Überdeckungen geben.

Algorithmus für Kanonische Überdeckung/1

Eine kanonische Überdeckung der Menge F von FDs kann folgendermaßen berechnet werden.

1. **Linksreduktion:** Führe für alle $X \rightarrow Y \in F$ eine Linksreduktion durch.
 - Linksreduktion für $X \rightarrow Y$: Überprüfe für alle einzelnen Attribute $A \in X$:

$$Y \subseteq \mathcal{AH}(F, X - A)$$

Falls dies gilt, ist A überflüssig und $X \rightarrow Y$ wird in F durch $(X - A) \rightarrow Y$ ersetzt:

$$F := F - \{X \rightarrow Y\} \cup \{(X - A) \rightarrow Y\}$$

2. **Rechtsreduktion:** Führe für alle (verbleibenden) $X \rightarrow Y \in F$ eine Rechtsreduktion durch.

- Rechtsreduktion für $X \rightarrow Y$: Überprüfe für alle $B \in Y$ ob

$$B \in \mathcal{AH}(F - \{X \rightarrow Y\} \cup \{X \rightarrow (Y - B)\}, X)$$

Falls dies gilt, ist B auf der rechten Seite überflüssig und kann eliminiert werden, d.h., $X \rightarrow Y$ wird in F durch $X \rightarrow (Y - B)$ ersetzt.

Algorithmus für Kanonische Überdeckung/2

3. **Entfernen von leeren Mengen:** Entferne alle FDs der Form $X \rightarrow \emptyset$, die möglicherweise durch die Rechtsreduktion entstanden sind.
4. **Vereinigung:** Fasse mittels der Vereinigungsregel FDs der Form $X \rightarrow Y_1, X \rightarrow Y_2, \dots, X \rightarrow Y_n$ zusammen zu $X \rightarrow (Y_1 \cup Y_2 \cup \dots \cup Y_n)$.

Eigenschaften des Algorithmus:

- Dieser Algorithmus erzeugt eine der möglichen kanonischen Überdeckungen.
- Je nach Ordnung der FDs können andere kanonische Überdeckungen herauskommen.

Integrierte Übung 5.4

Gegeben die Relation $R[A, B, C, D, E, F]$ mit der Menge $F = \{A \rightarrow BC, C \rightarrow DA, E \rightarrow ABC, F \rightarrow CD, CD \rightarrow BEF\}$ von FDs.

- Bestimmen Sie eine kanonische Überdeckung von F .
- Berechnen Sie die Attributhülle von A
- Bestimmen Sie alle Kandidatenschlüssel von R .

Inhalt

- 1 Richtlinien für Relationalen Datenbankentwurf
- 2 Funktionale Abhängigkeiten
- 3 Zerlegung von Relationen
- 4 Normalformen

Zerlegungen und deren Eigenschaften

- Zerlegung (decomposition) von Relationen
- Verlustlosigkeit (lossless join decomposition)
- Abhängigkeitsbewahrung (dependency preservation)

Relationale Entwurf mit Universalschema

- **Universalschema:** Schema einer Relation R , das *alle* Attribute der Datenbank enthält.
- **Zerlegung:** Schema einer Relation R in Schemas $Z = \{R_1, R_2, \dots, R_n\}$ zerlegen, sodass

$$sch(R) = sch(R_1) \cup sch(R_2) \cup \dots \cup sch(R_n)$$

- **Relationaler Entwurf** durch Dekomposition:
 1. Starte mit dem Universalschema R .
 2. Zerlege R so, dass Redundanzen vermieden werden.
- Bei der Zerlegung spielen **FDs eine wesentliche Rolle**.
- **Korrektheitskriterien** für Zerlegungen von R in $Z = \{R_1, R_2, \dots, R_n\}$:
 - Verlustlosigkeit: für jede Instanz muss R aus R_1, R_2, \dots, R_n rekonstruierbar sein.
 - Abhängigkeitsbewahrung: die FDs von R müssen auf R_1, R_2, \dots, R_n übertragbar sein.

Verlustlosigkeit

- **Verlustlosigkeit** (lossless join decomposition):
Eine Zerlegung von R mit FDs F in R_1, R_2, \dots, R_n ist genau dann verlustlos, wenn für jede Instanz von R die F erfüllt gilt:

$$\pi_{sch(R_1)}(R) \bowtie \pi_{sch(R_2)}(R) \bowtie \dots \bowtie \pi_{sch(R_n)}(R) = R$$

- **Beachte:** Das Wort “verlustlos” bezieht sich auf Information, nicht die Anzahl der Tupel. Im Gegenteil: Joins auf nicht verlustfreie Zerlegungen erzeugen zusätzliche (falsche) Tupel.
- **Satz:** R_1 und R_2 sind eine **verlustfreie Zerlegung** von R bezüglich der FDs F genau dann wenn
 - $(sch(R_1) \cap sch(R_2)) \rightarrow sch(R_1)$ ist in F^+ oder
 - $(sch(R_1) \cap sch(R_2)) \rightarrow sch(R_2)$ ist in F^+
- **Intuition:** Die Join-Attribute zwischen R_1 und R_2 sollen entweder für R_1 oder R_2 einen Schlüssel darstellen.

Integrierte Übung 5.5

Gegeben $R[A, B, C]$, $F = \{AB \rightarrow C, C \rightarrow B\}$, $R_1[A, C]$, $R_2[B, C]$.

- Ist $\{R_1, R_2\}$ eine verlustfrei Zerlegung von R ?
- Probiere die Zerlegung an $R = \{(\alpha, 0, a), (\beta, 2, b), (\gamma, 1, c), (\alpha, 2, b)\}$ aus.
- Was passiert mit der Zerlegung, wenn das Tupel $(\alpha, 2, b)$ durch $(\alpha, 2, c)$ ersetzt wird, sodass $C \rightarrow B$ nicht mehr erfüllt ist?

Abhängigkeitsbewahrung/1

- Geben eine Zerlegung $Z = \{R_1, R_2, \dots, R_n\}$ von R mit FDs F_R .
- Die **Einschränkung** F_{R_i} von F_R auf R_i enthält alle FDs $X \rightarrow Y \in F_R^+$ mit $(X \cup Y) \subseteq R_i$.
- **Abhängigkeitsbewahrung:** Eine Zerlegung $Z = \{R_1, R_2, \dots, R_n\}$ von R mit FDs F_R ist abhängigkeitsbewahrend genau dann wenn für die entsprechenden Einschränkungen F_{R_i} gilt:

$$F_R^+ = (F_{R_1} \cup F_{R_2} \cup \dots \cup F_{R_n})^+$$

Abhängigkeitsbewahrung/2

- **Intuition:** Bei abhängigkeitsbewahrender Zerlegung kann jede FD aus F **lokal** auf einer Relation R_i geprüft werden.
- **Praktische Bedeutung** der Abhängigkeitsbewahrung:
 - FDs müssen bei jeder Änderung der Datenbank geprüft werden.
 - Wenn FDs nicht auf einzelnen Relation R_i geprüft werden können, muss ein Join $R_i \bowtie R_j$ zur Prüfung durchgeführt werden.
 - Das ist in der Praxis viel zu teuer.

Integrierte Übung 5.6

Gegeben $R[A, B, C, D]$ und $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow D, D \rightarrow A, A \rightarrow D\}$. Ist die Zerlegung in $R1(A, B)$, $R2(B, C)$, und $R3(C, D)$ abhängigkeitsbewahrend?

Normalformen

Normalformen:

- First Normal Form (1NF)
- Second Normal Form (2NF)
- Third Normal Form (3NF)
- Boyce-Codd Normal Form (BCNF)

Höhere Normalformen (nicht behandelt):

- Fourth Normal Form (4NF)
- Fifth Normal Form (5NF)

Inhalt

- 1 Richtlinien für Relationalen Datenbankentwurf
- 2 Funktionale Abhängigkeiten
- 3 Zerlegung von Relationen
- 4 Normalformen

Normalisierung/1

- **Normalisierung:** Die Attribute eines (schlechten) Schemas einer Relation werden auf kleinere (gute) Schemas aufteilen, welche den Normalformen genügen.
- Die Normalisierung wurde von **Codd im Jahr 1972** eingeführt.
- Während des **Normalisierungsprozesses** werden eine Reihe von Tests auf einem Schema durchgeführt um zu überprüfen, ob sich das Schema in einer bestimmten Normalform befindet.
- Eine **normalisierte Datenbank** besteht aus *guten* Schemas.

Normalisierung/2

Übersicht über die Normalformen:

- 1NF: Attributwerte müssen atomar sein.
- 2NF, 3NF, BCNF: basieren auf Schlüsseln und FDs einer Relation
- 4NF: basieren auf Schlüsseln und mehrwertigen Abhängigkeiten (multi-valued dependencies, MVDs)
- 5NF: basieren auf Schlüsseln und Join Dependencies (JDs)

Weiters müssen beim relationalen Entwurf berücksichtigt werden:

- **Verlustlosigkeit** der entsprechenden Joins (sehr wichtig, darf niemals geopfert werden)
- **Abhängigkeitsbewahrung** der funktionalen Abhängigkeiten (kann unter Umständen aufgegeben werden)

Normalisierung/3

- In der Praxis wird normalisiert um ein Schema hoher Qualität zu garantieren.
- Der Normalisierungsprozess führt zu einem vertieften Verständnis der Relationen und Attribute.
- Datenbank-Designer brauchen nicht bis zur höchsten Normalform normalisieren:
 - es gibt einen Tread-off zwischen NF und Abfrage-Effizienz
 - normalerweise wird 3NF, BCNF oder 4NF ausgewählt

Normalisierung/4

- **Kontrollierte Redundanz:**
 - Redundanz, welche dem System bekannt ist
 - kontrollierte Redundanz ist gut
 - Beispiele: Fremdschlüssel, Indexe
- **Denormalisierung:**
 - Der Join mehrerer Relationen in einer höheren Normalform wird als Relation gespeichert.
 - Die Ergebnis-Relation befindet sich in einer niedrigeren Normalform, da Joins Normalformen zerstören.

Erste Normalform (1NF)/1

- **Verbietet:**
 - zusammengesetzte Attribute
 - mehrwertige Attribute
 - verschachtelte Relationen: Attribute, deren Wert für jedes Tupel eine Relation ist
- 1NF wird oft als **Teil der Definition** einer Relation gesehen.
- **Beispiel:** Folgende Instanz der Relation *Fachbereiche* ist nicht in 1NF (mehrwertiges Attribut):

Fachbereiche			
FName	FNum	LeiterSVN	Standorte
Research	5	334455	{ Salzburg, Wien, Linz }
Administration	4	987654	{ Innsbruck }
Headquarters	1	888666	{ Linz }

Erste Normalform (1NF)/2

- **Abhilfe** um 1NF zu erhalten:
 - zusammengesetzte Attribute: jeder Teil wird ein eigenes Attribut
 - mehrwertige Attribute: neues Tupel für jeden Wert des mehrwertigen Attributs erzeugen
 - geschachtelte Relationen: neues Tupel für jedes Tupel der geschachtelten Relation erzeugen
- **Beispiel:** *Fachbereiche* in 1NF gebracht.

Fachbereiche_1NF

FName	FNum	LeiterSVN	Standort
Research	5	334455	Salzburg
Research	5	334455	Wien
Research	5	334455	Linz
Administration	4	987654	Innsbruck
Headquarters	1	888666	Linz

Zweite Normalform (2NF)/1

- **Zweite Normalform (2NF):** Eine Relation R befindet sich in der zweiten Normalform (2NF) genau dann wenn sie sich in 1NF befindet und jedes Nicht-Schlüssel Attribut voll funktional abhängig von allen Kandidatenschlüsseln ist.
- **Nicht-Schlüssel Attribut:** Attribut, das nicht Teil eines Kandidatenschlüssels (inklusive Primärschlüssel) ist.
- Ein Attribut A ist **voll funktional abhängig** vom Kandidatenschlüssel K ($K \xrightarrow{\bullet} A$), wenn es keine echte Teilmenge $X \subset K$ gibt, sodass $X \rightarrow A$:

$$K \xrightarrow{\bullet} A \Leftrightarrow K \rightarrow A \wedge \forall X \subset K : X \not\rightarrow A$$

- **Intuition:** 2NF ist verletzt, wenn mehrere Entitäten in einer einzigen Relation modelliert werden.

Zweite Normalform (2NF)/2

- **Beispiel:** Folgende Relation ist nicht in 2NF:

AngProj

SVN	PNum	Stunden	AName	PName	POrt
1234	1	32.5	Schmidt	ProductX	Salzburg
1234	2	7.5	Schmidt	ProductY	Wien
6688	3	40.5	Mair	ProductZ	Linz
4567	1	20.0	Huber	ProductX	Salzburg
4567	2	20.0	Huber	ProductY	Wien
3334	2	10.0	Wong	ProductY	Wien
3334	3	10.0	Wong	ProductZ	Linz
3334	10	10.0	Wong	Computerization	Innsbruck
3334	20	10.0	Wong	Reorganization	Linz

- **Warum ist *AngProj* nicht in 2NF?**
 - Kandidatenschlüssel ist $\{SVN, PNum\}$, von dem aber nur *Stunden* voll funktional abhängig ist.
 - *SVN* ist ein Teilschlüssel der *AName* bestimmt.
 - *PNum* ist ein Teilschlüssel der *PName* und *POrt* bestimmt.

Zweite Normalform (2NF)/3

- **Abhilfe** um 2NF zu erhalten:
 - Erzeuge eine neue Relation für jeden Teilschlüssel mit seinen abhängigen Attributen.
 - Behalte eine Relation mit dem ursprünglichen Schlüssel und allen vom Schlüssel voll funktional abhängigen Attribute.
- **Beispiel:** *AngProj*[*SVN*, *PNum*, *Stunden*, *AName*, *PName*, *POrt*]
 - FDs:
 - $\{SVN, PNum\} \rightarrow Stunden, SVN \rightarrow AName, PNum \rightarrow \{PName, POrt\}$
 - $\{SVN, PNum\}$ ist einziger Kandidatenschlüssel.
 - *SVN* und *PNum* sind Teilschlüssel mit abhängigen Attributen.

2NF Normalisierung von *AngProj*:

- *AngProj*1(*SVN*, *AName*)
- *AngProj*2(*PNum*, *PName*, *POrt*)
- *AngProj*3(*SVN*, *PNum*, *Stunden*)

Integrierte Übung 5.7

Betrachte $R[A, B, C]$ mit $F = \{A \rightarrow BC, B \rightarrow C\}$. Ist R in 2NF? Ist R ein gutes Schema?

Dritte Normalform (3NF)/1

- **Dritte Normalform (3NF)**: Eine Relation R befindet sich in 3NF genau dann wenn sie sich in 1NF befindet und für alle FDs $X \rightarrow Y \in F^+$ mindestens eine der folgenden Bedingungen gilt:
 - $X \rightarrow Y$ ist trivial (d.h. $Y \subseteq X$)
 - X ist ein Superschlüssel von R
 - jedes Attribut $A \in Y$ ist in einem Kandidatenschlüssel von R enthalten
- **Intuition**: 3NF verbietet transitive Abhängigkeiten.
- **$3NF \subset 2NF$** : eine Relation in 3NF ist auch in 2NF

Dritte Normalform (3NF)/2

- Die folgende Relation ist in 2NF aber nicht in 3NF:

AngFB

AName	SVN	Jahrgang	Adresse	FNum	FName	LeiterSVN
Schmidt	1234	1965	Linz	5	Research	2345
Schmidt	2345	1965	Linz	5	Research	2345
Wong	6688	1968	Linz	4	Admin	4567
Zelaya	4567	1941	Linz	4	Admin	4567
Borg	3334	1937	Dallas	4	Admin	4567

- Funktionale Abhängigkeiten:
 - $fd1 : SVN \rightarrow sch(AngFB)$
 - $fd2 : FNum \rightarrow \{FName, LeiterSVN\}$
- $fd2$ erfüllt keine der drei Bedingungen für 3NF:
 - $fd2$ ist nicht trivial
 - $FNum$ ist keine Superschlüssel von $AngFB$
 - $FName$ ist in keinem Kandidatenschlüssel enthalten

Dritte Normalform (3NF)/3

- $AngFB[AName, SVN, Jahrgang, Adresse, FNum, FName, LeiterSVN]$ mit FDs:
 - Primärschlüssel SVN: $SVN \rightarrow sch(AngFB)$
 - $FNum \rightarrow \{FName, LeiterSVN\}$
- $\{FName, LeiterSVN\}$ sind transitiv abhängig vom Schlüssel SVN:
 - $SVN \rightarrow FNum, FNum \rightarrow FName$
 - $SVN \rightarrow FNum, FNum \rightarrow LeiterSVN$
- Zerlegung von $AngFB$:
 - $Ang[AName, SVN, Jahrgang, Adresse, FNum]$
 - $FB[FNum, FName, LeiterSVN]$

Ang und FB sind Relationen in 3NF.

Boyce-Codd NormalForm/1

- **Boyce-Codd Normal Form (BCNF):** Eine Relation R ist in BCNF genau dann wenn sie in 1NF ist und für alle $X \rightarrow Y \in F^+$ mindestens eine der folgenden Bedingungen gilt:
 - $X \rightarrow Y$ ist trivial (d.h. $Y \subseteq X$)
 - X ist ein Superschlüssel von R
- **BCNF \subset 3NF:** Eine Relation in BCNF ist auch in 3NF.
- **Beispiel:** Folgende Relation ist in 3NF aber nicht in BCNF:

Lernen

Stud	Kurs	Buch
Schmidt	Data Structures	Bertram
Schmidt	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz
Gale	Data Structures	Horowitz

Syntheselgorithmen zur Zerlegung in 3NF/1

- **Syntheselgorithmen:** Zerlegt das Schema einer Relation R mit funktionalen Abhängigkeiten F in die Schemas R_1, R_2, \dots, R_n mit folgenden Eigenschaften:
 - alle R_i ($1 \leq i \leq n$) sind in 3NF
 - die Zerlegung ist **verlustfrei**
 - die Zerlegung ist **abhängigkeitsbewahrend**

Syntheselgorithmen zur Zerlegung in 3NF/2

Relation R mit funktionalen Abhängigkeiten F in 3NF zerlegen:

1. Bestimme kanonische Überdeckung F_c zu F .
2. Für jede funktionale Abhängigkeit $X \rightarrow Y \in F_c$:
 - a. Kreiere eine Relation R_X mit dem Schema $X \cup Y$.
 - b. Ordne R_X die FDs $F_X = \{X' \rightarrow Y' \in F_c \mid X' \cup Y' \subseteq R_X\}$ zu.
3. Falls keine neue Relation R_X einen Kandidatenschlüssel von R bezüglich F_c enthält, wähle einen Kandidatenschlüssel $K \subseteq sch(R)$ aus:
 - a. Erzeuge eine Relation R_K mit Schema K .
 - b. Die FDs von R_K sind $F_K = \emptyset$.
4. Eliminiere R und alle neu erzeugten Relationen R_i die in einer anderen Relation R_j enthalten sind, d.h., $sch(R_i) \subseteq sch(R_j)$.

Boyce-Codd Normal Form/2

Lernen

Stud	Kurs	Buch
Schmidt	Data Structures	Bertram
Schmidt	Data Management	Martin
Hall	Compilers	Hoffman
Brown	Data Structures	Horowitz
Gale	Data Structures	Horowitz

- **FDs in der Relation *Lernen*:**
 - fd1: $\{Stud, Kurs\} \rightarrow Buch$
 - fd2: $Buch \rightarrow Kurs$
- ***Lernen* ist in 3NF:**
 - fd1: $\{Stud, Kurs\}$ ist Kandidatenschlüssel (d.h. auch Superschlüssel)
 - fd2: $Kurs$ ist im Kandidatenschlüssel enthalten
- ***Lernen* ist nicht in BCNF:**
 - fd2 ist nicht trivial und $Buch$ ist kein Superschlüssel

Boyce-Codd Normal Form/3

- Zerlegung von *Lernen* in $KB[Kurs, Buch]$ und $SB[Stud, Buch]$:

KB		SB	
Kurs	Buch	Stud	Buch
Data Structures	Bertram	Schmidt	Bertram
Data Management	Martin	Schmidt	Martin
Compilers	Hoffman	Hall	Hoffman
Data Structures	Horowitz	Brown	Horowitz
		Gale	Horowitz

- FDs in den Relationen *KB* und *SB*:
 - $KB : Buch \rightarrow Kurs$
 - SB : nur triviale Anhängigkeiten
 - $\{Kurs, Stud\} \rightarrow Buch$ von *Lernen* ist verloren gegangen.

Dekompositionsalgorithmus zur Zerlegung in BCNF/1

- **Dekompositionsalgorithmus**: Zerlegt das Schema einer Relation *R* mit funktionalen Abhängigkeiten *F* in die Schemas R_1, R_2, \dots, R_n mit folgende Eigenschaften:
 - alle R_i ($1 \leq i \leq n$) sind in **BCNF**
 - die Zerlegung ist **verlustfrei**
- Die Zerlegung in BCNF ist in manchen Fällen **nicht abhängigkeitsbewahrend**.

Dekompositionsalgorithmus zur Zerlegung in BCNF/2

Relation *R* mit funktionalen Abhängigkeiten *F* in BCNF zerlegen:

1. $Z = \{R\}$
2. Solange es ein $R_i \in Z$ gibt, sodass R_i nicht in BCNF ist:
 - a. Finde eine FD $X \rightarrow Y$ für R_i , sodass:
 - $X \cap Y = \emptyset$, d.h. keine (teilweise) triviale Anhängigkeit
 - $X \not\rightarrow R_i$, d.h. X ist kein Superschlüssel von R_i
 - b. Zerlege R_i in zwei Relationen R_{i1} und R_{i2} mit den Schemas
 - $sch(R_{i1}) = X \cup Y$
 - $sch(R_{i2}) = R_i - Y$
 - c. Ersetze R_i durch R_{i1} und R_{i2} in Z .

Integrierte Übung 5.8

Gegeben $R[Kurs, Professor, Zeit, Raum, Student, Note]$ und folgende FDs:

- $K \rightarrow P$ jeder Kurs hat nur einen Professor
- $RZ \rightarrow K$ ein Kurs in einem Raum zu einer bestimmten Zeit
- $PZ \rightarrow R$ ein Professor kann zu einer bestimmten Zeit nur in einem Raum lehren
- $SK \rightarrow N$ Studenten bekommen für jeden Kurs nur eine Note
- $SZ \rightarrow R$ Studenten können zu einer bestimmten Zeit nur in einem Raum sein

Zerlege das Schema verlustlos in BCNF.

Zusammenfassung Normalformen

- Die **gebräuchlichsten Normalformen** sind:
 - 1NF: atomare Attribute
 - 2NF: jede Relation entspricht einer eigenen Entität
 - 3NF: keine transitiven Abhängigkeiten
 - BCNF: nur Schlüsselabhängigkeiten erlaubt
- Wenn eine Relation in **BCNF** ist, gibt es **keine Redundanz** aufgrund funktionaler Abhängigkeiten mehr.
- Für die Normalformen gilt:
$$\text{BCNF} \subset \text{3NF} \subset \text{2NF} \subset \text{1NF}$$
- Jede Relation lässt sich **verlustlos** bis zu **BCNF** zerlegen.
- Jede Relation lässt sich **abhängigkeitsbewahrend** bis zu **3NF** zerlegen.
- Bei der Zerlegung in BCNF können FDs verloren gehen.