



UNIT 0x07

NORMALFORMEN I

FUNKTIONALE ABHÄNGIGKEITEN

Motivation/Erinnerung

Aufgabe

- Entwicklung einer 'guten' Datenbank zu einem realen Problem (z.B. Mini-Welt), d.h. u.a. vollständig, korrekt, minimal, aber auch redundanzfrei und effizient...

Vorgehen

- Konsolidierung und Konzeptueller Entwurf in Form eines ER-Modells ✓
- Überführung in eine 'gute' Datenbank... dazu benötigen wir:
 - Implementationsentwurf, z.B. mit Tabellen und Relationen
→ Relationales Modell ✓
 - Mathematisches Modell der Datenbankoperationen, z.B. zur Anfrageoptimierung
→ Relationale Algebra ✓
 - Gütemaß für den Implementationsentwurf
→ Normalformen

Qualität eines Datenbankentwurfs

Anforderungen an unsere Artikel-Sammlung, u.a.

- Widerspruchsfreie Speicherung von Daten jeder Art
- Erinnerung
erster Entwurf:

Nr (EAN,ISBN)	Zeitschrift	Verlags- gründung	Themen
...aa11	Heise - c't - 11/18	1949	S.15 C++, S.24 C#
...bb22	Heise - c't - 12/18	1949	S.12 Python, S.29 C#
...cc33	Heise - ix - 08/18	1949	S.9 RAID, S.23 gpio
...dd44	Computec - buffed - 08/18	1989	S.11 WoW
...ee55	Webedia - GameStar - 08/18	2007	S.13 LoL, S.20 WoW

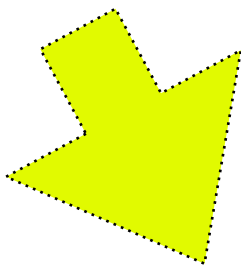
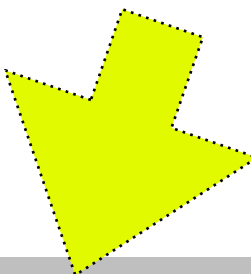
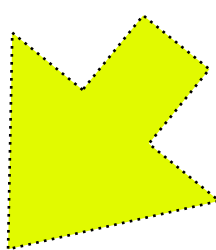
Q&A

- Was war noch gleich die Kritik?

Qualität eines Datenbankentwurfs

”Besserer” Entwurf, Kernideen

- Redundanzen verringern
- Sachverhalte separieren



Nr (N)	Verlag (V)	Magazin (M)	Ausgabe (A)
aa11	Heise	c't	11/18
bb22	Heise	c't	12/18
cc33	Heise	iX	08/18
dd44	Computec	buffed	08/18
ee55	Webedia	GameStar	08/18

Nr (EAN,ISBN)	Zeitschrift	Verlagsgründung	Themen
...aa11	Heise - c't - 11/18	1949	S.15 C++, S.24 C#
...bb22	Heise - c't - 12/18	1949	S.12 Python, S.29 C#
...cc33	Heise - ix - 08/18	1949	S.9 RAID, S.23 gpio
...dd44	Computec - buffed - 08/18	1989	S.11 WoW
...ee55	Webedia - GameStar - 08/18	2007	S.13 LoL, S.20 WoW

Nr (N)	Seite (S)	Thema (T)
aa11	15	C++
aa11	24	C#
bb22	12	Python
bb22	29	C#
cc33	9	RAID
cc33	23	gpio
dd44	11	WoW

Verlag (V)	Gründung (G)
Heise	1949
Computec	1989
Webedia	2007

Q&A

- Wie ist die Qualität eines Entwurfs zu messen?
- Wie erhält man systematisch eine Verbesserung?
- Kann man direkt einen guten Entwurf erstellen?

Qualität eines Datenbankentwurfs

Wie ist die Qualität eines Entwurfs zu messen?

- Gütekriterien, genannt *Normalformen*

Wie erhält man systematisch eine Verbesserung?

- Definitionen: Funktionale Abhängigkeiten, Attributhüllen, Kanonische Überdeckung
- Mathematische Werkzeuge: Armstrong-Axiome
- Algorithmen: Synthese- und Dekompositionsalgorithmus

Kann man 'direkt' einen guten Entwurf erstellen?

- *Spoiler: Ja!* Aber dafür müssen die Kriterien verstanden sein...

Attribute und Funktionale Abhängigkeit

Beispiel Artikel-Datenbank

- Ähnlich wie bei Attributen im ER-Diagramm gehen wir von einer sinnvollen Aufteilung, d.h. separierten Sachverhalten, aus.
- Die Attribute/Spalten seien wie folgt abgekürzt:

Nr (N)	Verlag (V)	Magazin (M)	Ausgabe (A)	Gründung (G)	Seite (S)	Thema (T)
aa11	Heise	c't	11/18	1949	15	C++
aa11	Heise	c't	11/18	1949	24	C#
bb22	Heise	c't	12/18	1949	12	Pytho

N=Nr, V=Verlag, M=Magazin, G=Gründung,
A=Ausgabe, S=Seite, T=Thema

kurz: $R = \{N, V, M, A, S, T, G\}$

Q&A

- Welche Attribute kann man aus anderen ableiten? Bzw.
- Für welche Attribute gilt: Kennt man dieses, kennt man auch jenes?

Attribute und Funktionale Abhängigkeit

Beispiel Artikel-Datenbank

Nr (N)	Verlag (V)	Magazin (M)	Ausgabe (A)	Gründung (G)	Seite (S)	Thema (T)
aa11	Heise	c't	11/18	1949	15	C++
aa11	Heise	c't	11/18	1949	24	C#
bb22	Heise	c't	12/18	1949	12	Python

- Abhängige Information/Attribute:
 - Wenn die (ISBN)Nr. (N) feststeht, dann sind der Verlag (V) mit Gründung (G), das Magazin (M) und die Ausgabe (A) klar.
 - Aber das Thema (T) steht erst fest, wenn z.B. die Nr. (N) und Seite (S) genannt sind.
 - Umgekehrt: Wenn der Verlag (V) bekannt ist, dann auch die Gründung (G).
 - Und: Wenn das Magazin (M) genannt wird, dann leitet sich aus dieser Information der Verlag (V) ab.
- (Funktionale) Abhängigkeiten in Kurzform: $N \rightarrow VMAG$, $NS \rightarrow T$, $V \rightarrow G$, $M \rightarrow V$.




$X \rightarrow Y$ bedeutet: Wenn X bekannt ist, dann auch Y.

Attribute und Funktionale Abhängigkeit

Ausgangssituation der folgenden Untersuchungen

- Gegeben ist einer Relation R mit (vereinfachtem) Schema { A, B, C,... }, und
- eine Menge von Abhängigkeiten der Form $X \rightarrow Y$, d.h. kennt man X, dann auch Y, wobei X und Y jeweils für eins der Attribute des Schemas stehen.



Die konkrete Bedeutung von X bzw. Y ist *nicht relevant*, man stelle sich den Verlag oder ein Thema vor!
Es geht vielmehr um die *Zusammenhänge* der Informationen.

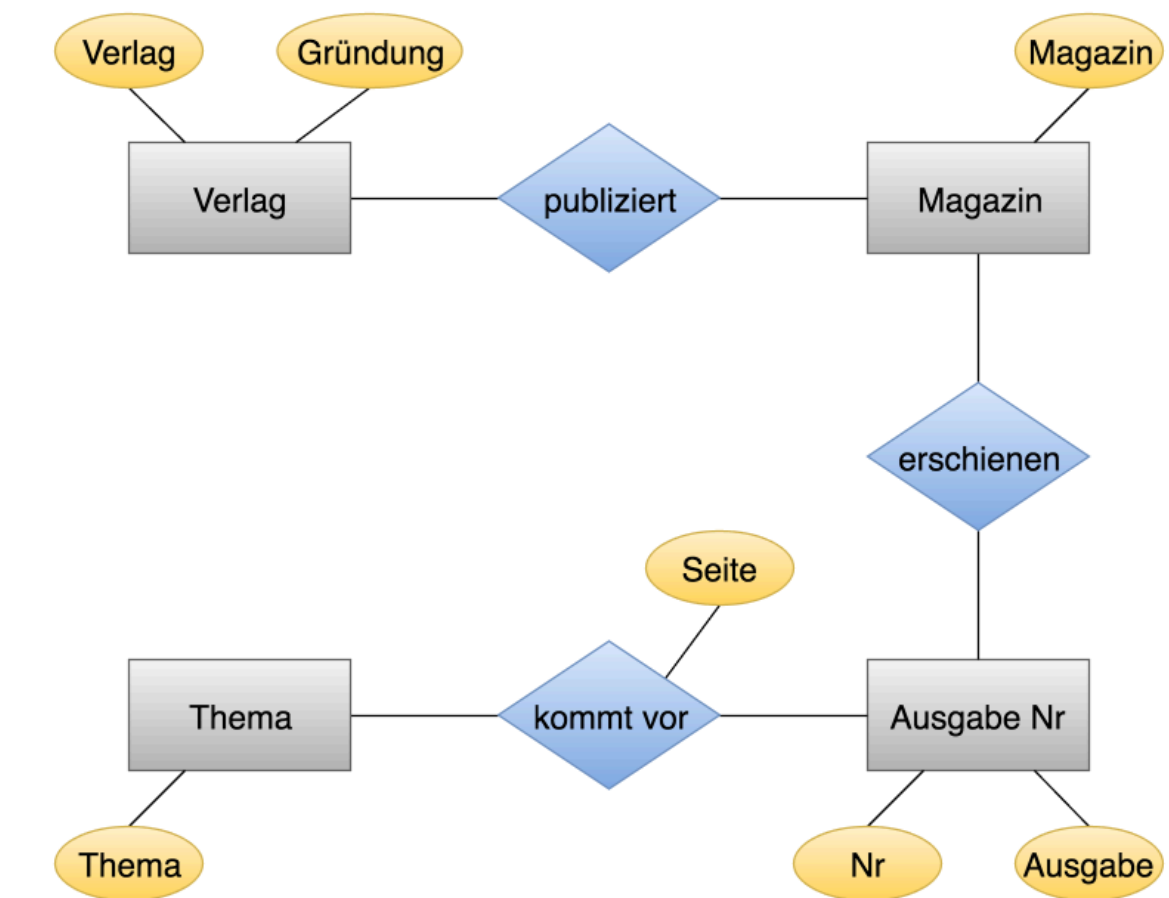
Q&A

- Aus welchen Attributen folgen alle anderen?
- Kann man die Menge der Abhängigkeiten vereinfachen?
- Wie sehen die Relationen bzw. Tabellen am Ende aus?

Wiederholung/Ziel

Zwei Ansätze

- Zu Beginn haben wir die Artikeldatenbank (intuitiv) in verschiedene Tabellen überführt, nur mit Überlegungen zu Informationsseparation und Redundanzreduktion. Es gab keine Mini-Welt, sondern nur diesen Ansatz.
- Eine Mini-Welt können wir in ein ER-Diagramm und dann in einen Implementationsentwurf überführen.



Die Überlegungen zu Normalformen zeigen im Folgenden, dass beide Ansätze systematisch zum gleichen Ergebnis kommen – einer 'guten' Datenbank.



DB->normalizer (TUM)

- Tool, um diese Überführung zu demonstrieren und zum Üben! (Link am Ende).

DB->normalizer (TUM)

Die 'Relation' sind die verwendeten Attribute und die 'FDs' die Abhängigkeiten aus unserem Beispiel. Details und Definitionen folgen, jetzt erstmal der Effekt:

Relation eingeben

NVMASTG

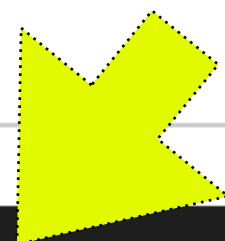
FDs/MVDs eingeben

N->VMAG

V->G


M->V

NS->T



Ergebnis anzeigen

Quiz

 Schema speichern

Schema laden ▲

DB->normalizer (TUM)

Eingabe

Relation

R:={AGMNSTV}

FDs

N->AGMV
V->G
M->V
NS->T



Gütekriterien

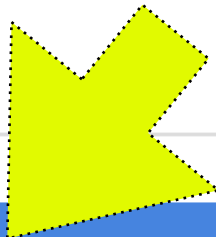
Ergebnis

Kandidatschlüssel

{NS}

Normalformen

1NF ✓ 2NF ⚡ 3NF ⚡ BCNF ⚡ 4NF ⚡



DB->normalizer (TUM)

➤ Kanonische Überdeckung

Algorithmen

1 Linksreduktion

N->AGMV
V->G
M->V
NS->T

2 Rechtsreduktion

N->AM
V->G
M->V
NS->T

3 $\alpha \rightarrow \emptyset$ entfernen

N->AM
V->G
M->V
NS->T

4 FDs zusammenfassen

N->AM
V->G
M->V
NS->T

Ergebnis



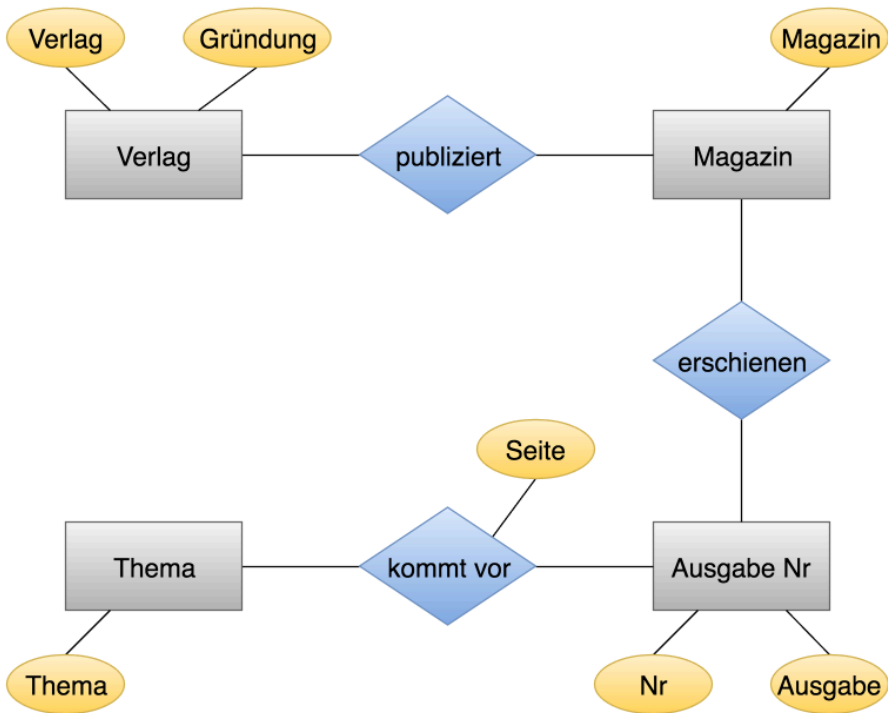
Nachdenken

DB->normalizer (TUM)

- Die Darstellung fokussiert weniger die IDs und Relationen sondern primär die Aufteilung der Informationen.
- Eine Abhängigkeit, etwa $V \rightarrow G$, ist dann realisiert, wenn beide Attribute in einer Tabelle stehen (mit geeigneten Schüsseln).

[V,G]

Verlag (V)	Gründung (G)
Heise	1949
Computec	1989
Webedia	2007



[M,V]

V-ID	Magazin (M)
101	c't
101	iX
102	buffed
103	GameStar

T-ID	Thema (T)
201	C++
202	C#
203	Python
204	RAID
205	gpio
206	WoW
207	LoL

N-ID	T-ID	Seite (S)
401	201	15
401	202	24
402	203	12
402	202	29
403	204	9
403	205	23
404	206	11
405	207	13
405	206	20

M-ID	Nr (N)	Ausgabe (A)
301	aa11	11/18
301	bb22	12/18
302	cc33	08/18
303	dd44	08/18
304	ee55	08/18

[N,M,A]

[N,S,T]

4 FDs zusammenfa

Algorithmen

N->AM

V->G

M->V

NS->T

Funktionale Abhängigkeit

Definition

- Sei $R=\{a_1,\dots,a_n\}$ ein vereinfachtes Schema, $\alpha \subseteq \text{ident}(R)$, $\beta \subseteq \text{ident}(R)$ und $D \subseteq \text{dom}(R)$.
Dann ist β **funktional abhängig von** α genau dann, wenn für alle zulässigen D gilt:

$$\forall s, t \in D : s.\alpha = t.\alpha \Rightarrow s.\beta = t.\beta,$$

in Zeichen $\alpha \rightarrow \beta$. Hierbei meint $s.\alpha = t.\alpha$ die Gleichheit in allen Attributen $a \in \alpha$.

$\alpha \rightarrow \beta$ heißt, für alle Tupel s, t mit gleichen α -Attributen gilt, dass auch ihre β -Attribute gleich sind. Die Werte der Attribute aus der Attributmenge α bestimmen also eindeutig die Werte der Attribute aus der Attributmenge β .

Bisher Verlag \rightarrow Gründung: Kennt man den Verlag (α) so auch das Gründungsjahr (β).
Obige Definition: bei gleichem Verlag ($s.\alpha$) ist das Gründungsjahr ($t.\beta$) gleich.



Funktionale Abhängigkeit

Anmerkungen zur Definition

- *Funktionale Abhängigkeit*, oder FD für *Functional Dependencies*, beschreibt eine Eigenschaft der *realen Welt*, nicht der gerade vorliegenden Daten. Diese dürfen dazu aber natürlich nicht im Widerspruch stehen!

Beispiel: Betrachte zu $R=\{A,B,C\}$ und den funktionalen Abhängigkeiten $f_1: \{A\} \rightarrow \{B,C\}$ und $f_2: \{C\} \rightarrow \{B\}$ diese Datenmenge D . Dann ist D verträglich mit f_1 , aber f_2 kann nicht gelten, da einmal $b1$ und einmal $b2$ aus $c1$ folgt. Weiter kann nicht einfach $\{B\} \rightarrow \{A\}$ geschlossen werden, nur weil D gerade diese Ausprägung hat.

A	B	C
a1	b1	c1
a2	b3	c2
a3	b2	c1
a1	b1	c1

$D \subseteq \text{dom}(R)$

- Statt ' $\{A\} \rightarrow \{B,C\}$ ' schreibt man auch kurz ' $A \rightarrow BC$ ', statt ' $R=\{A,B,C\}$ ' auch ' $R=ABC$ '.

**$AB \rightarrow CD$: Wenn A und B bekannt, dann auch C und D .
Nur Kenntnis von A oder B *alleine* sagt nichts aus!**



Funktionale Abhängigkeit

Definition

- Sei $R=\{a_1, \dots, a_n\}$ ein vereinfachtes Schema und $\alpha \rightarrow \beta$ eine funktionale Abhängigkeit. Dann ist β **voll funktional abhängig von** α wenn

$$\forall \gamma \in \mathcal{P}(\alpha) - \alpha: \alpha - \gamma \not\rightarrow \beta$$

in Zeichen: $\alpha \twoheadrightarrow \beta$. Das bedeutet, α ist minimal.


Volle funktionale Abhängigkeit bedeutet, dass α minimal ist, man also kein Attribut aus α weglassen kann, ohne dass man die funktionale Abhängigkeit verliert.



Schlüssel

Definition

- Sei $R=\{a_1, \dots, a_n\}$ ein vereinfachtes Schema und $\alpha \subseteq \text{ident}(R)$. Dann ist
 - α ein **Superschlüssel**, falls $\alpha \rightarrow \text{ident}(R)$, und
 - α ein **Schlüsselkandidat**, falls $\alpha \dot{\rightarrow} \text{ident}(R)$.
- Ein **Primärschlüssel** ist ein ausgesuchter/festgelegter *Schlüsselkandidat*.
- Ein Attribut $a \in \text{ident}(R)$ heißt **prim**, falls a Attribut *eines* Schlüsselkandidaten von R ist, sonst **nicht prim**.




Funktionale Abhängigkeit aller Attribute vom Primärschlüssel bedeutet nichts anderes als dass man die gesamte Entität kennt, wenn man den Schlüssel hat... in völliger Übereinstimmung mit unserer Idee einer *id.*

Schlüssel

Anmerkungen zur Definition

- Es macht Sinn, aus den möglichen Schlüsselkandidaten einer Relation R genau *einen* Primärschlüssel festzulegen, da Verweise auf Tupel aus R über sog. *Fremdschlüssel* in anderen Relationen realisiert werden, also Schlüsselattribute dieses Primärschlüssels.
- Im DBMS werden Primärschlüssel explizit gekennzeichnet und führen dazu, dass sie in der Tabelle nicht `null` sein oder doppelt vorkommen dürfen. Daraus ergibt sich insbesondere, dass in der Tabelle keine zwei Tupel identisch sind (Eindeutigkeit).
- *Nicht* jedes α mit $\alpha \rightarrow \text{ident}(R)$ ist ein *Schlüsselkandidat*, denn α muss nicht minimal sein! Insbesondere ist $\alpha = \text{ident}(R)$ der triviale Superschlüssel.



Achtung: Es kann beispielsweise Schlüsselkandidaten AB und BCD geben, auch wenn BCD 'länger' als AB ist... die *Minimalität* bedeutet hier, dass man weder A oder B aus AB noch B,C oder D aus BCD weglassen kann, ohne die Schlüsseleigenschaft zu verlieren!

Kurzer Halt

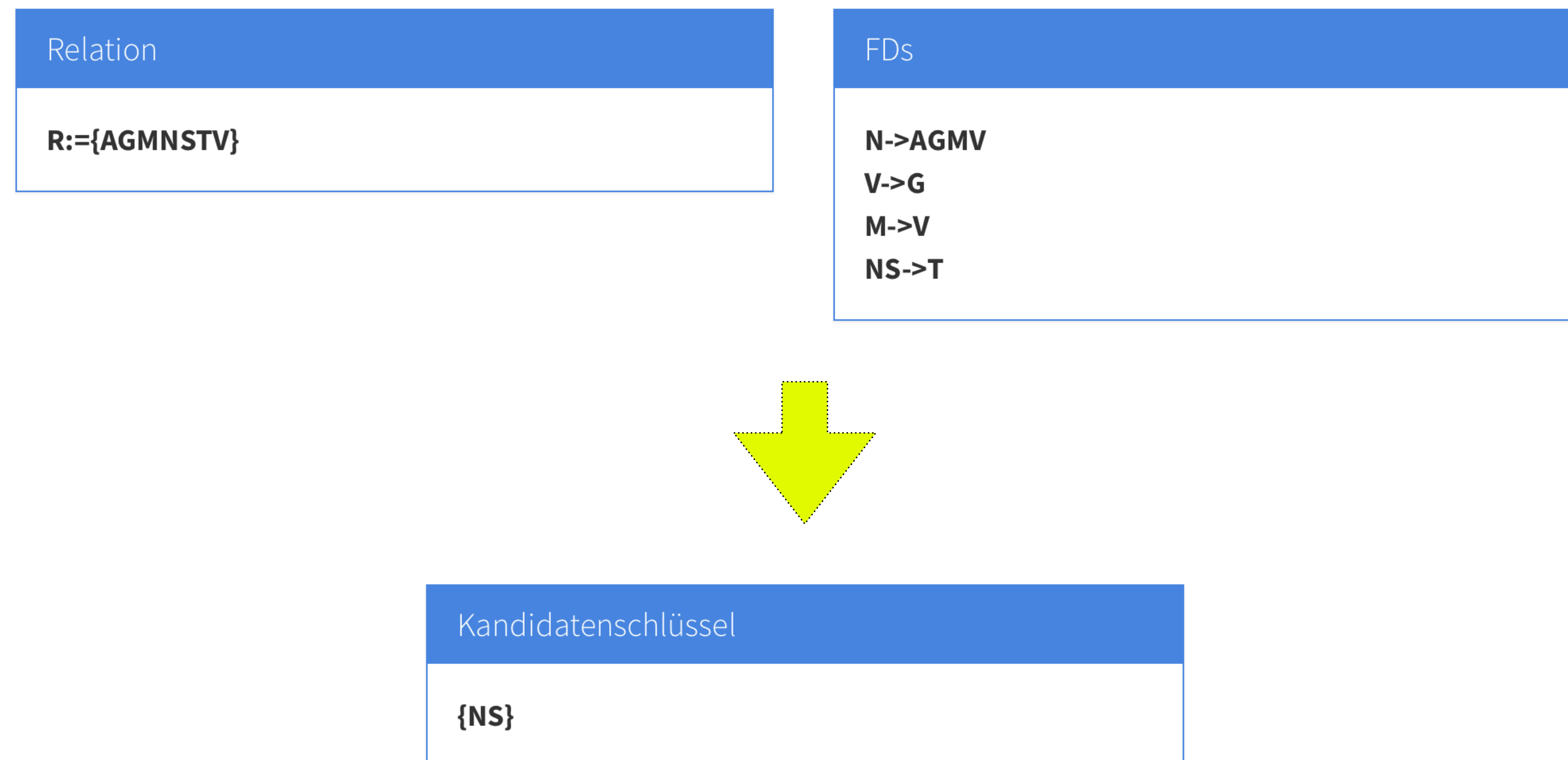


Die bisherigen Definitionen sind wichtig, ohne sie versteht man den Rest nicht mehr!

- A oder B stehen für reale Attribute wie der Verlag (V) oder die Gründung (G) in der Artikeldatenbank.
- $R=ABCD$, $FDs=\{ A \rightarrow B, AB \rightarrow CD \}$.
- Superschlüssel, Schlüsselkandidat, Primärschlüssel, prim, minimal.

Kurzer Halt

Nochmal ein Blick auf den Normalizer mit den Definitionen im Kopf:



Armstrong-Axiome

Motivation

- Gegeben sei ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$

Q&A

- Gilt $A \rightarrow C$?
- Aus welchen Attributen folgen alle anderen Attribute?
Anders: Wie lauten die Schlüsselkandidaten?
- Kann man die FDs 'vereinfachen'?
Anders: Kann man FDs weglassen oder 'kürzen' bei gleicher 'Aussagekraft'?

Armstrong-Axiome

Motivation

- Gegeben sei ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$

Gilt $A \rightarrow C$?

- Ja, denn ist A bekannt, dann auch B (wg. $A \rightarrow B$) und dann auch C (wg. $B \rightarrow C$); genannt *Transitivität*
- Es gibt offenbar Regeln, nach denen man aus den FDs oben weitere ableiten kann – die sog. **Armstrong-Axiome** (folgen).

Armstrong-Axiome

Motivation

- Gegeben sei ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$

Wie lauten die Schlüsselkandidaten?

- Wir suchen grundsätzlich einzelne Attribute oder Attributkombinationen, aus denen alle anderen Attribute folgen und wo wir nichts weglassen können.
- Dazu starten wir mit einem Attribut, z.B. A, und sammeln zusammen, was folgt. Danach probieren wir weitere Attribute und ggf. Kombinationen, bis alle Schlüsselkandidaten bekannt sind.
- Dieses qualifizierte Raten kann man auf unterschiedliche Weise und optimiert durchführen, aber am Ende muss man ein paar Kombinationen ausprobieren...

Armstrong-Axiome

Motivation

- Gegeben sei ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$

Wie lauten die Schlüsselkandidaten? Fortsetzung

- Zu A: Aus A folgt B ($A \rightarrow B$), und dann C ($B \rightarrow C$). Da wir mit A auch B kennen, folgen C und D ($AB \rightarrow CD$) und insgesamt $A \rightarrow ABCD$. Also ist A ein (erster) Schlüsselkandidat.
- B: Aus B folgt nur C, nicht A oder D.
- C, D: Weder aus C noch aus D folgen weitere Attribute.
- Weitere Kombinationen mit A, etwa ABC, sind Superschlüssel aber nicht Schlüsselkandidat, denn man kann alles ausser A weglassen und somit sind sie nicht minimal.
- Weitere Kombinationen ohne A, also BC, CD und BCD, führen nie zu A.

A ist einziger Schlüsselkandidat!



Armstrong-Axiome

Motivation

- Gegeben sei ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$

Kann man die FDs 'vereinfachen'?

- Ja, z.B. die 'kürzeren' FDs
 - $\{ A \rightarrow B, B \rightarrow C, A \rightarrow CD \}$ (da $A \rightarrow B$ gilt, ist $A \rightarrow CD$ 'so gut wie' $AB \rightarrow CD$)
 - $\{ A \rightarrow B, B \rightarrow C, A \rightarrow D \}$ (da $A \rightarrow B$ und $B \rightarrow C$ gilt, ist $A \rightarrow D$ 'so gut wie' $AB \rightarrow CD$)
 - $\{ A \rightarrow BD, B \rightarrow C \}$ (da zuvor $A \rightarrow B$ und $A \rightarrow D$ gilt, auch $A \rightarrow BD$)sagen das gleiche aus.

Q&A

- Wie ist 'so gut wie' definiert?

Armstrong-Axiome

Mit Hilfe der **Armstrong-Axiome** lassen sich funktionale Abhängigkeiten aus einer Menge von FDs ableiten (wie zuvor bei der Transitivität). Die folgenden drei Regeln reichen aus, um alle funktionalen Abhängigkeiten herzuleiten:

Definition

- Sei $R=\{a_1, \dots, a_n\}$ ein vereinfachtes Schema und $\alpha, \beta, \gamma \subseteq \text{ident}(R)$. Die **Armstrong-Axiome** sind gegeben durch:
 - **Reflexivität**: für alle $\beta \subseteq \alpha$ gilt: $\alpha \rightarrow \beta$ (triviale Abhängigkeit).
 - **Transitivität**: gilt $\alpha \rightarrow \beta$ und $\beta \rightarrow \gamma$, so auch $\alpha \rightarrow \gamma$.
 - **Anreicherung/Verstärkung**: gilt $\alpha \rightarrow \beta$, so auch $\alpha\gamma \rightarrow \beta\gamma$.
- Strenggenommen sind es keine 'Axiome' im mathematischen Sinne, denn sie lassen sich aus der Definition der funktionalen Abhängigkeit ableiten – heissen aber so.

Armstrong-Axiome

Aus den gegebenen Armstrong-Axiomen können weitere Regeln abgeleitet werden, um ggf. Herleitungen zu vereinfachen.

Definition

- Sei $R=\{a_1,\dots,a_n\}$ ein vereinfachtes Schema und $\alpha,\beta,\gamma,\delta\subseteq\text{ident}(R)$. Es gilt weiter:
 - **Vereinigung:** gilt $\alpha\rightarrow\beta$ und $\alpha\rightarrow\gamma$, so auch $\alpha\rightarrow\beta\gamma$.
 - **Dekomposition:** gilt $\alpha\rightarrow\beta\gamma$, so auch $\alpha\rightarrow\beta$ und $\alpha\rightarrow\gamma$.
 - **Pseudotransitivität:** gilt $\alpha\rightarrow\beta$ und $\beta\gamma\rightarrow\delta$, so auch $\alpha\gamma\rightarrow\delta$.

Armstrong-Axiome

Erinnerung

- Gegeben war ein Schema $R=\{A,B,C,D\}$ mit $FDs=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$.

Wie lauten die Schlüsselkandidaten?

- Wir haben A hergenommen und überlegt, was aus den FDs folgt. Das ist aber nichts anderes, als die Regeln, also die Armstrong-Axiome, anzuwenden.
- Wir haben hier die sog. *Attributhülle von A bzgl. FD* berechnet (folgt sofort).

Kann man die FDs 'vereinfachen'?

- Wir haben uns Regeln überlegt, die 'so gut wie' andere sind – ebenfalls Armstrong-Axiome. Das führt auf die sog. *Kanonische Überdeckung* (folgt in Teil II).

Attributhülle

Definition

- Zu einem Schema R und einer Menge FD (manchmal F) funktionaler Abhängigkeiten bestimmt die **Attributhülle**
 - ♦ $\text{AttrHülle}(\text{FD}, \alpha)$, oder α^+ , wenn FD klar, alle Attribute, die bzgl. FD funktional abhängig von α sind.
- *Achtung:* Algorithmus terminiert spätestens, wenn Ergebnismenge alle Attribute enthält. In diesem Fall ist α sogar Superschlüssel.
- *Achtung:* Initiale Menge ist immer α selbst (Reflexivität).

```
// ----- AttrHülle(F,  $\alpha$ ) -----  
// Eingabe:  
//   F: Menge von funktionalen Abhängigkeiten  
//    $\alpha$ : Menge von Attributen  
// Ausgabe:  
//   transitiver Abschluss von  $\alpha$  bzgl.  $F \rightarrow \alpha^+$   
  
AttrHülle(F,  $\alpha$ ) {  
    Erg = { $\alpha$ } ; ErgAlt =  $\emptyset$   
    while (Erg  $\neq$  ErgAlt) {  
        ErgAlt = Erg;  
        foreach (  $\beta \rightarrow \gamma$  in F)  
            if ( $\beta \subseteq \text{Erg}$ ) then Erg = Erg  $\cup$  { $\gamma$ }  
    }  
    return Erg  
}
```

Attributhülle

Beispiel

Schema $R=\{A,B,C,D\}$, $FD=\{ A \rightarrow B, B \rightarrow C, AB \rightarrow CD \}$, gesucht sind:

- $\text{AttrH\ddot{u}lle}(FD,A)$: $A \rightarrow A \rightarrow AB \rightarrow ABC \rightarrow ABCD$ fertig

wg. Start $A \rightarrow B \quad B \rightarrow C \quad AB \rightarrow CD$

d.h. $A^+ = ABCD = \text{ident}(R)$.

- $\text{AttrH\ddot{u}lle}(FD,B)$: $B \rightarrow B \rightarrow BC$ fertig

wg. Start $B \rightarrow C$

d.h. $B^+ = BC \neq \text{ident}(R)$.

- $\text{AttrH\ddot{u}lle}(FD,C)$: $C \rightarrow C$ fertig

d.h. $C^+ = C$.

- $\text{AttrH\ddot{u}lle}(FD,D)$: $D \rightarrow D$ fertig

d.h. $D^+ = D$.

Q&A

- Wie schreibt man das auf, falls mal jemand fragt... ?

Attributhülle

Beispiel

Schema $R=\{A,B,C,D\}$, $FD=\{ A\rightarrow BCD, CD\rightarrow A \}$, gesucht sind alle Attributhüllen:

- $\text{AttrHülle}(FD,A)$: $A\rightarrow A\rightarrow ABCD$ fertig, also $A^+ = ABCD = \text{ident}(R)$ (Schlüsselkandidat).
- AB^+ , AC^+ , AD^+ alles Superschlüssel.
- B^+ , C^+ , D^+ , BC^+ , BD^+ alle trivial und insbes. $\neq \text{ident}(R)$.
- $\text{AttrHülle}(FD,CD)$: $CD\rightarrow CD\rightarrow ACD\rightarrow ABCD$, also $CD^+ = \text{ident}(R)$ (Schlüsselkandidat).

Der Algorithmus zur Bestimmung der Attributhülle liefert, auf alle Kombinationen angewandt, *alle* Schlüsselkandidaten.

Achtung: *A und* CD sind Schlüsselkandidaten, obwohl CD 'länger' als A ist... aber beide sind minimal.



Attributhülle

Bestimmung der Schlüsselkandidaten

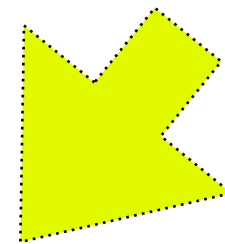
- Es gibt im Wesentlichen zwei Möglichkeiten:
 - Alle Möglichkeiten ausprobieren.
 - Mit $\text{ident}(R)$ anfangen und sinnvoll streichen (folgt bei *Kanonischer Überdeckung*).
- Problem ist, dass *alle Möglichkeiten* sehr viele sein können... aber:
 - Man kann geschickt anfangen, indem man zuerst alle Attribute berücksichtigt, die nicht gefolgert werden können (falls es sie gibt, Beispiel folgt).
 - Im Laufe der Berechnung einer Attributhülle ergeben sich Attributkombinationen, die z.B. einen Schlüsselkandidaten enthalten – dann ist man auch fertig, oder die man zuvor schon berücksichtigt hat.



Insgesamt sieht es nach mehr Arbeit aus, als es ist.

Attributhülle

Beispiel



Schema $R=\{A,B,C,D\}$, $FD=\{ A \rightarrow BC \}$, gesucht sind alle Schlüsselkandidaten:

- Grundidee: Wir benötigen $\alpha \rightarrow \text{ident}(R)$, d.h. *alle* Attribute müssen gefolgt werden.

Q&A

- Wie soll D in die Attributhülle gelangen?

- Kurz gesagt: Gar nicht, wenn es nicht schon drin ist.

Alle Attribute, die *nicht* auf einer rechten Seite vorkommen, *müssen* in die Schlüsselkandidaten!



- Es kann passieren, dass alle auf einer rechten Vorkommen... das ist dann Pech.

Attributhülle

Fortsetzung Beispiel

Schema $R=\{A,B,C,D\}$, $FD=\{A \rightarrow BC\}$, gesucht sind alle Schlüsselkandidaten:

- Es ist $D \in \text{ident}(R)$, aber D kommt auf keiner rechten Seite vor. D.h. alle Kombinationen *ohne* D , insbes. A, B, C , können kein Schlüsselkandidat sein.
- Der einzige mögliche Schlüsselkandidat mit nur einem Attribut ist D , aber man sieht, dass das wegen $D^+ = D$ nicht reicht.
- Der nächstbeste Kandidat ist folglich AD : $AD^+ = AD \rightarrow ABCD = \text{ident}(R)$, also Schlüsselkandidat.
- Da in FD nur aus A etwas folgt, ist man dann hier auch fertig.
- Insgesamt ist somit AD einziger Schlüsselkandidat und A und D sind *prim*, B und C sind *nicht prim*.

Attributhülle

Beispiel

Schema $R=\{A,B,C,D\}$, $FD=\{ A\rightarrow BCD, CD\rightarrow A \}$, gesucht sind alle Schlüsselkandidaten. Man kann wie folgt argumentieren (statt alles blind durchzuprobieren):

- Zunächst: Kein Attribut fehlt auf der rechten Seite – Schade.
- A ist offensichtlicher Schlüsselkandidat, CD aufgrund der Transitivität ebenso.
- Man sieht in FD, dass weder aus B, noch aus C oder D alleine $ABCD=\text{ident}(R)$ folgt, also B,C und C keine Schlüsselkandidaten.
- Jede Obermenge von A, also AB, AC etc., ist Superschlüssel, aber nicht minimal, da A reicht. Gleiches gilt für Obermengen von CD und es bleibt nichts mehr übrig zu untersuchen.
- Insgesamt sind A und CD Schlüsselkandidaten; A,C,D prim, B nicht prim.

Attributhülle

Beispiel

Schema $R=\{A,B,C\}$, $FD=\{ A\rightarrow B, B\rightarrow C, C\rightarrow A \}$, gesucht sind alle Schlüsselkandidaten:

- Zunächst: Kein Attribut fehlt auf der rechten Seite – Schade.
- Man sieht: $A^+=B^+=C^+=ABC=\text{ident}(R)$.
- Mehr Kombinationen müssen nicht probiert werden, da alle Kombinationen Obermengen von A, B oder C sind.
- Insgesamt sind A, B und C Schlüsselkandidaten und alle Attribute *prim*.

Zusammenfassung

Roter Faden

- Armstrong-Axiome und Attributhülle: Grundlage für Umformungen der FDs.
- Kanonische Überdeckung, Synthesealgorithmus, Dekompositionsalgorithmus, Gütekriterien Normalformen folgen in Teil II und III.
- Link zum Normalizer: <https://normalizer.db.in.tum.de/index.py>