

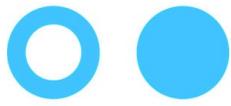
Teil II

Von Daten und ihren Modellen

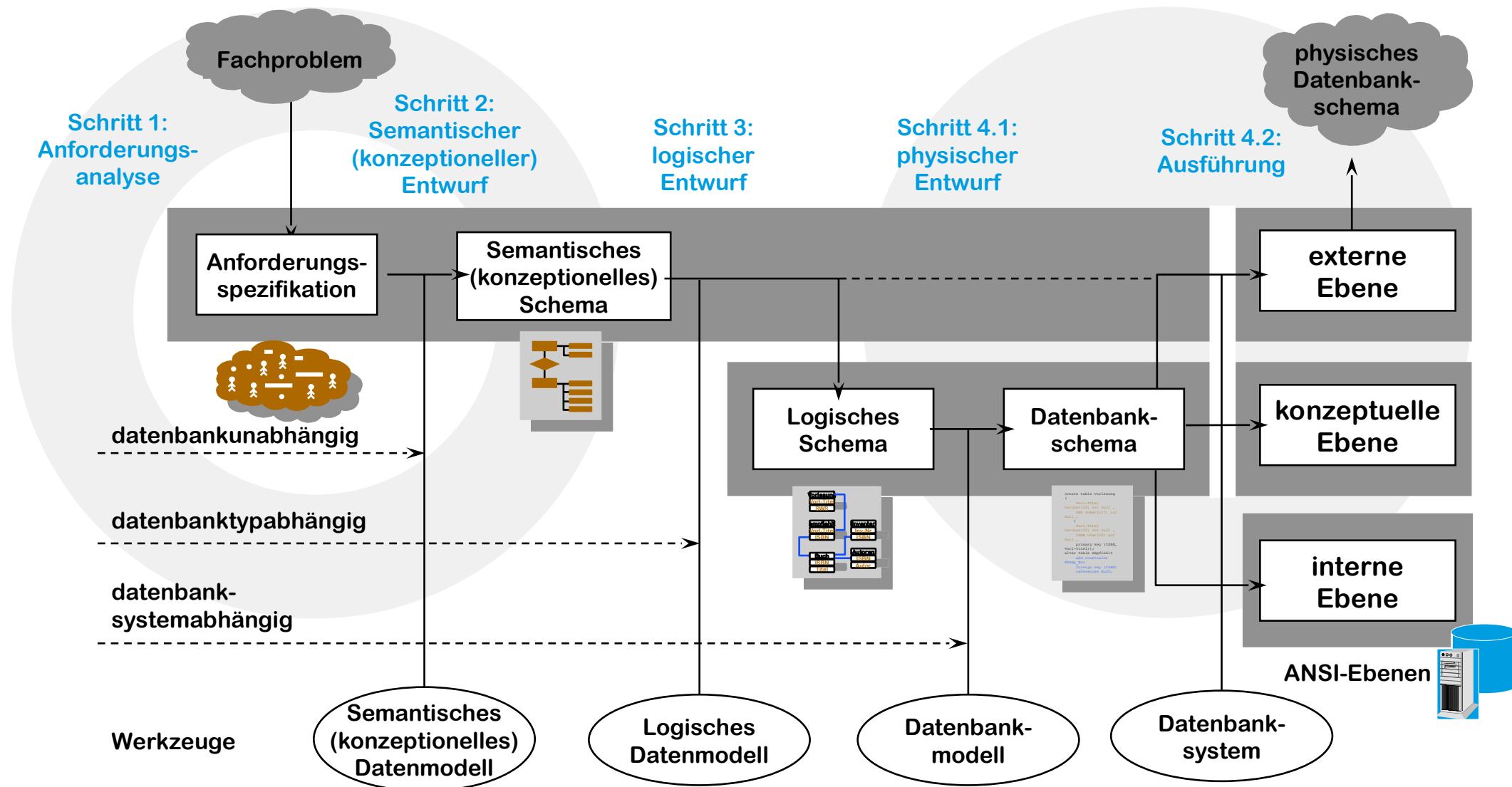
Robert Hartmann (SoSe 2024)

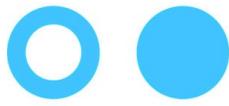
basierend auf Folien von
Prof. Dr. Harm Knolle

Fachbereich Informatik
Hochschule Bonn-Rhein-Sieg



- Vom Fachproblem zur Datenbank -





- Miniwelt vs. Semantisches Datenschema -

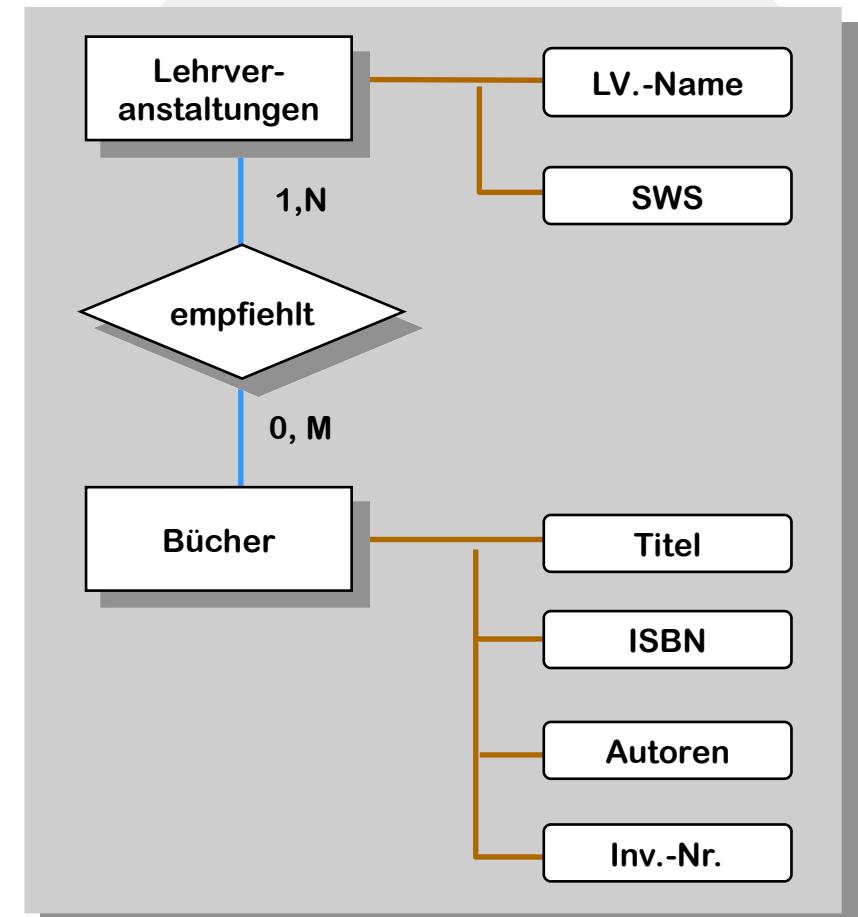
Miniwelt (Diskursbereich) mittels

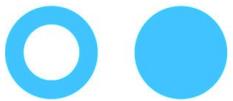
- freier Notation



Semantisches (konzeptionelles) Schema mittels

- „Entity-Relationship“-Datenmodell





- Kapitel 4 - Logische Datenmodelle -

Inhalt

0 - Vorbemerkungen

Teil I - Von EDV-Anwendungen und Ihren Anforderungen

1 - Einführung

Teil II - Von Daten und ihren Modellen

2 - Prozess des Datenbankentwurfs

3 - Semantische Datenmodelle

4 - Logische Datenmodelle

5 - Datenbankmodelle

6 - Datenanfrage und Datenänderung

Teil III - Von Datenbanken und ihren Systemen

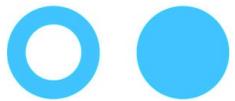
7 - Datenbanksysteme

8 - Speicherstrukturen

9 - Ausblick

Überblick

- ◆ Einführung
- ◆ Historische Datenmodelle
- ◆ Relationale Modell
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema



- Logische Datenmodelle -

Ziel

- ♦ Erstellung eines relationalen Schemas
- ♦ Transformation eines ER-Schemas in ein relationales Schema

Hilfsmittel

- ♦ Relationales Modell
- ♦ Theoretische Grundlagen der Normalisierung

Literatur

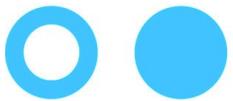
- ♦ KeEi15
 - Kapitel 3 „Das relationale Modell“
 - bis einschl. 3.3
 - Kapitel 5 „Datenintegrität“
 - bis einschl. 5.3
 - Kapitel 6 „Relationale Entwurfstheorie“

Inhalt

- ♦ Einführung
- ♦ Historische Datenmodelle
- ♦ Relationale Modell
- ♦ Normalisierung
- ♦ Abbildung eines ER- auf relationales Schema

Literatur

- ♦ Ku15
 - Kapitel 3: „Relationales Datenmodell“
 - bis einschl. 3.4.5.7
- ♦ SSH18
 - Kapitel 5: „Relationenmodell und Relationenalgebra“
 - bis einschl. 5.1.3
 - Kapitel 7: „Relationaler Datenbankentwurf“
 - bis einschl. 7.3



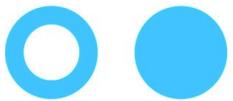
- Einführung (I) -

Semantisches Schema

- ◆ entwurfstechnische Darstellung der Miniwelt (einer Fachabteilung) des Unternehmens
- ◆ in der Regel graphische Beschreibung mit den Konzepten eines semantisch reichen Datenmodells (z.B. Entity Relationship Modell (ERM))
- ◆ beinhaltet typmäßige, jedoch keine wertmäßigen Ausprägungen der Realität
- ◆ fungiert als Schnittstelle zwischen Fachabteilung und EDV
- ◆ basiert auf eindeutigen, mit den Fachabteilungen eines Unternehmens festgelegten und für die EDV verbindlichen Fachbegriffen
- ◆ ist unabhängig von der Auswahl eines konkreten Datenbanksystems
- ◆ bildet die Grundlage für den logischen Entwurf der Datenbank

Logisches Schema

- ◆ vom Typ des Datenbanksystems abhängige Darstellung der Miniwelt (einer Fachabteilung) des Unternehmens
- ◆ in der Regel graphische oder formale Beschreibung mit wenigen Konzepten eines logischen Datenmodells (z.B. Relationale Modell)
- ◆ beinhaltet typmäßige, jedoch keine wertmäßigen Ausprägungen der Realität
- ◆ fungiert als Schnittstelle zwischen Anwendungsentwicklung und Datenbankadministration
- ◆ häufig (teil-)automatisierte Ableitung aus einem semantischen Schema möglich
- ◆ ist abhängig vom Typ der Auswahl konkreter Datenbanksysteme
- ◆ bildet die Grundlage für den systemspezifischen Entwurf der Datenbank



- Einführung (II) -

Bindegliedfunktion

- ◆ Datenbanksystemhersteller orientieren sich zur Definition und Manipulation der Daten in der Datenbank an bestimmten Datenmodellen (, halten sich aber nicht immer vollständig daran)
- ◆ logische Datenmodelle verstehen sich somit als Bindeglied und formale Rahmen für die Umsetzung eines semantischen Schemas in ein konkretes Datenbankschema
- ◆ logische Datenmodelle sind daher "idealisierte" Formen bestimmter Datenbankmodelle bestimmter Hersteller
- ◆ Verzicht auf systemspezifische Feinheiten, Beschränkung auf grundsätzliche Modellierungskonzepte bestimmter Zielsysteme
- ◆ derzeit werden von den kommerziell angebotenen Datenbanksystemen vier grundsätzlich unterschiedliche Stoßrichtungen unterstützt

Hierarchische Datenmodelle

- ◆ älteste Datenmodell mit stark abnehmender Bedeutung aber noch im Betrieb

Netzwerk Datenmodelle

- ◆ in den 70er Jahren stark favorisiert, aber heute nahezu bedeutungslos

Relationale Datenmodell

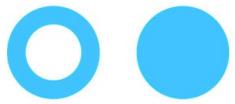
- ◆ derzeit größte praktische Bedeutung

Objekt-orientierte Datenmodelle

- ◆ wurde vielfach als Nachfolger der relationalen Datenmodelle gehandelt
- ◆ praktische Bedeutung für Anwendungen mit sehr speziellen Anforderungen der Modellierung
- ◆ erfolgreich sind objekt-relationale Datenmodelle

NoSQL Datenmodelle

- ◆ oft schemilos
 - ◆ starke Nähe zur Implementierung d. Anwendung
- NoSQL = Not only SQL



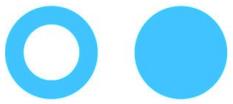
- Historische Datenmodelle -

Inhalt

- ◆ Einführung
- ◆ **Historische Datenmodelle**
- ◆ Relationale Modell
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

- ◆ Hierarchisches Modell
- ◆ Netzwerkmodell



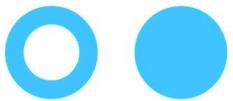
- Hierarchisches Modell -

Inhalt

- ◆ Einführung
- ◆ Historische Datenmodelle
 - Hierarchisches Modell
 - Netzwerkmodell
- ◆ Relationale Modell
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

- ◆ Historie
- ◆ Begriffe
- ◆ Ausprägung



- Historie -

Ursprung

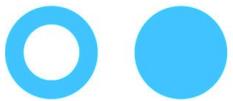
- ◆ 1968 als Datenmodell des Systems IMS (Information Management System) von IBM
- ◆ kommerziell erfolgreichste Datenmodell der ersten Generation
- ◆ viele Datenbestände heutiger Systeme aus den 70er Jahren sind hierarchisch organisiert (und werden das sicherlich auch noch einige Jahre ...)

Idee

- ◆ Daten stehen in den Knoten einer Hierarchie
- ◆ Zugriffe erfolgen traversierend und ausprogrammiert: gehen zum ersten Knoten, dann zum nächsten, vorherigen ...

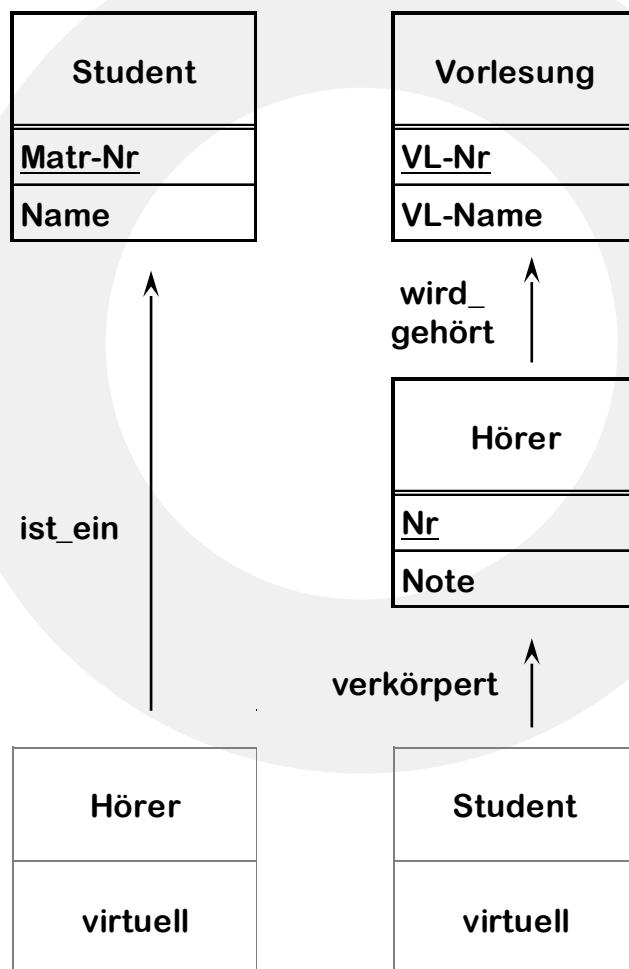
Gründe für das Aussterben

- ◆ komplexe Schemata, da nur 1:N-Beziehungen erlaubt sind
 - N:M-Beziehungen lassen sich nicht direkt implementieren
- ◆ kryptische, keine deskriptive Anfragesprache
- ◆ Datenmodell setzt viel Programmier- und Datenbankerfahrungen voraus
- ◆ rasante Entwicklung des wesentlich eleganteren relationalen Modells



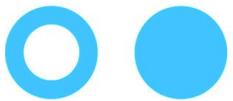
- Begriffe -

Schema



Konzepte

- ◆ **Segment Typ**
 - entspricht dem Entity-Typ im ER-Modell
 - Segment ist Ausprägung eines Segment Typs (entspricht Entity im ER-Modell)
- ◆ **Hierarchie**
 - baumartige Verknüpfung von Segmenttypen oder Segmenten
 - ein Nachfolger kann nur einen Vorgänger haben
- ◆ **Virtueller Segment Typ**
 - nur scheinbar vorhandener Record Typ
 - Zeiger auf einen korrespondierenden Record Typen einer anderen Hierarchie
 - zur Realisierung von N:M-Beziehungen
 - Beziehungen zwischen Hierarchien



- Hierarchisches Datenmodell -

System

- IMS (Information Management System) von IBM

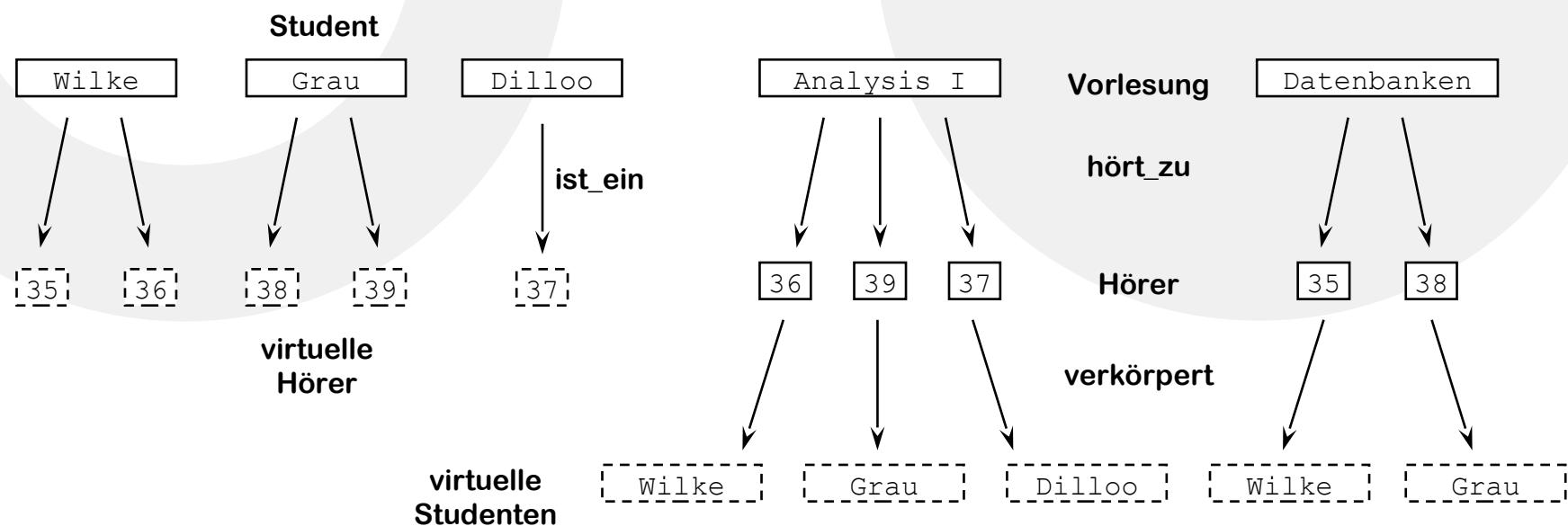
Sprache

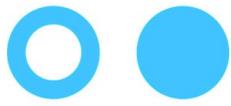
- DL/I (Data Language/I) von IBM
- basiert auf hierarchische Anordnung einzelner Datenelemente (Segmente)

- N:M-Beziehungen werden indirekt über logisch redundante Datenelemente realisiert (Zeiger)

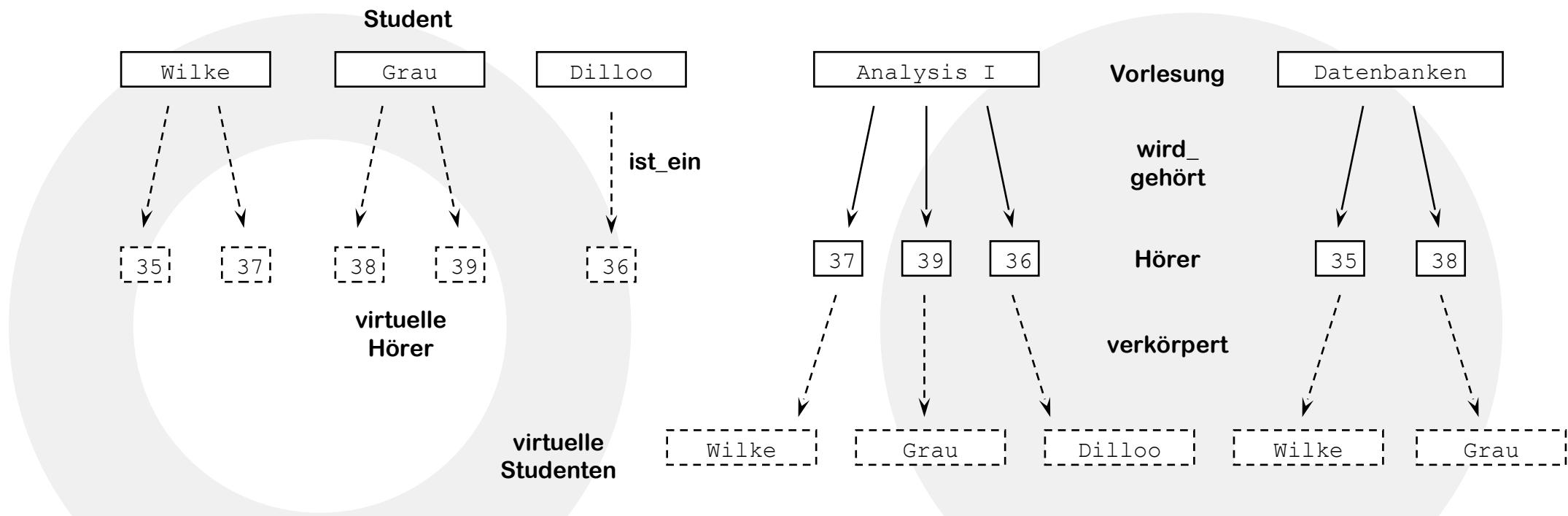
Anfragesprache

- Festlegung einer Position in einer Hierarchie typgleicher Segmente
- segmentorientiert traversierend (von oben nach unten, von links nach rechts)





- Ausprägung -

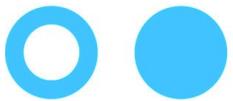


Realisierung

- "Wald" von Hierarchien
- Verkettung der Hierarchien durch virtuelle Segmente (Zeiger)

Nachteile

- Anwender muss Speicherstruktur kennen und benutzen, um in den Hierarchien traversieren zu können
- Verletzung der physischen und logischen Datenunabhängigkeit



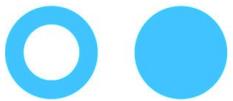
- Netzwerkmodell -

Inhalt

- ◆ Einführung
- ◆ Historische Datenmodelle
 - Hierarchisches Modell
 - Netzwerkmodell
- ◆ Relationale Modell
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

- ◆ Historie
- ◆ Begriffe
- ◆ Ausprägung



- Historie -

Netzwerkdatenmodell

- geht auf den 1969 von der DBTG (Data Base Task Group) auf der COnference on DAta SYstems Language spezifizierten CODASYL Vorschlag zurück
- eingesetzt u.a. als Datenmodell für das System UDS von Siemens
- wichtiger Meilenstein und Rahmen für die Entwicklung heutiger Datenbanksysteme

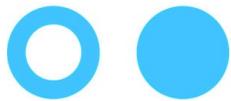
Idee

- Daten stehen in Knoten eines Netzes verknüpfter Beziehungen
- ein Knoten kann an mehrerer Beziehungen teilnehmen (im Gegensatz zur Hierarchie)
- direkte Implementierung von N:M-Beziehungen möglich

- Zugriffe erfolgen navigierend und ausprogrammiert:
 - Positionieren des ersten Knotens,
 - dann durchlaufen der von diesem Knoten im Rahmen der ausgewählten Beziehung unmittelbar abhängigen Knoten

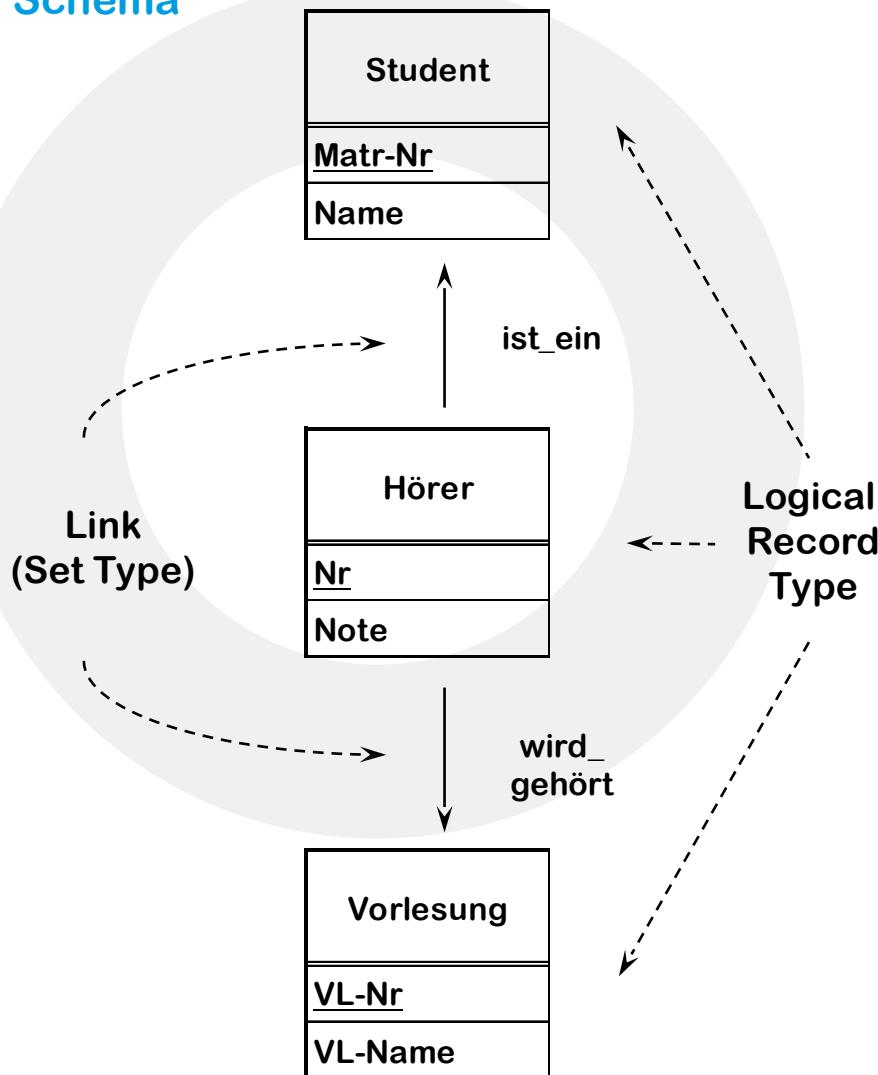
Gründe für das Aussterben

- sehr komplexes Modell
- Beteiligung vieler Einrichtungen am Vorschlag; daher erscheinen viele Konzepte als zusammenhangslos
- starker Bezug zur Sprache COBOL
- keine deskriptive Anfragesprache
- Vermengung physischer und logischer Konzepte (fehlende Datenunabhängigkeit)
- rasante Entwicklung des wesentlich eleganteren relationalen Modells



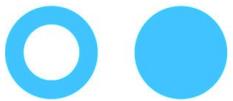
- Begriffe -

Schema



Konzepte

- ♦ **Logischer Record Typ**
 - entspricht dem Entity-Typ im ER-Modell
 - Liste mit Attributen
 - Record ist Ausprägung eines logischen Record Typen (entspricht Entity im ER-Modell)
- ♦ **Link (Set-Typ)**
 - entspricht 1:N-Beziehungstyp im ER-Modell
 - Record Typ mit der mehrfachen Kardinalität ist Besitzer des Set Typs
 - Record Typ mit der einfachen Kardinalität ist Teilnehmer des Set Typs
 - N:M-Beziehung, da ein Record Typ zu mehreren Set Typen gehören kann
 - Set entspricht Menge eines Eigentümer Records mit seinen Teilnehmer Records



- Netzwerk Datenmodell -

System

- z.B. UDS von Siemens

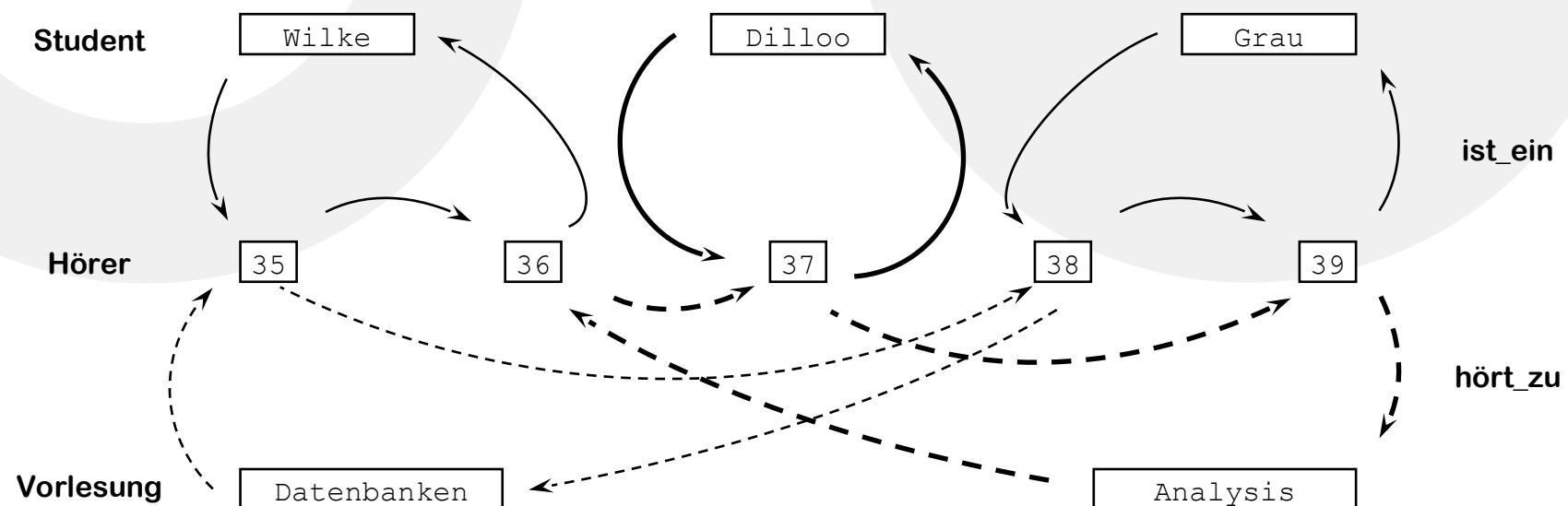
Sprache

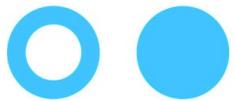
- CODASYL DBTG Language
 - COnference On DATA SYstem Languages
 - Data Base Task Group

- basiert auf netzartige Anordnung von Daten-elementen (Records), die über 1:N-Bezie-hungen (Sets) miteinander verbunden sind
- N:M-Beziehung direkt über mehrere 1:N-Beziehungen darstellbar

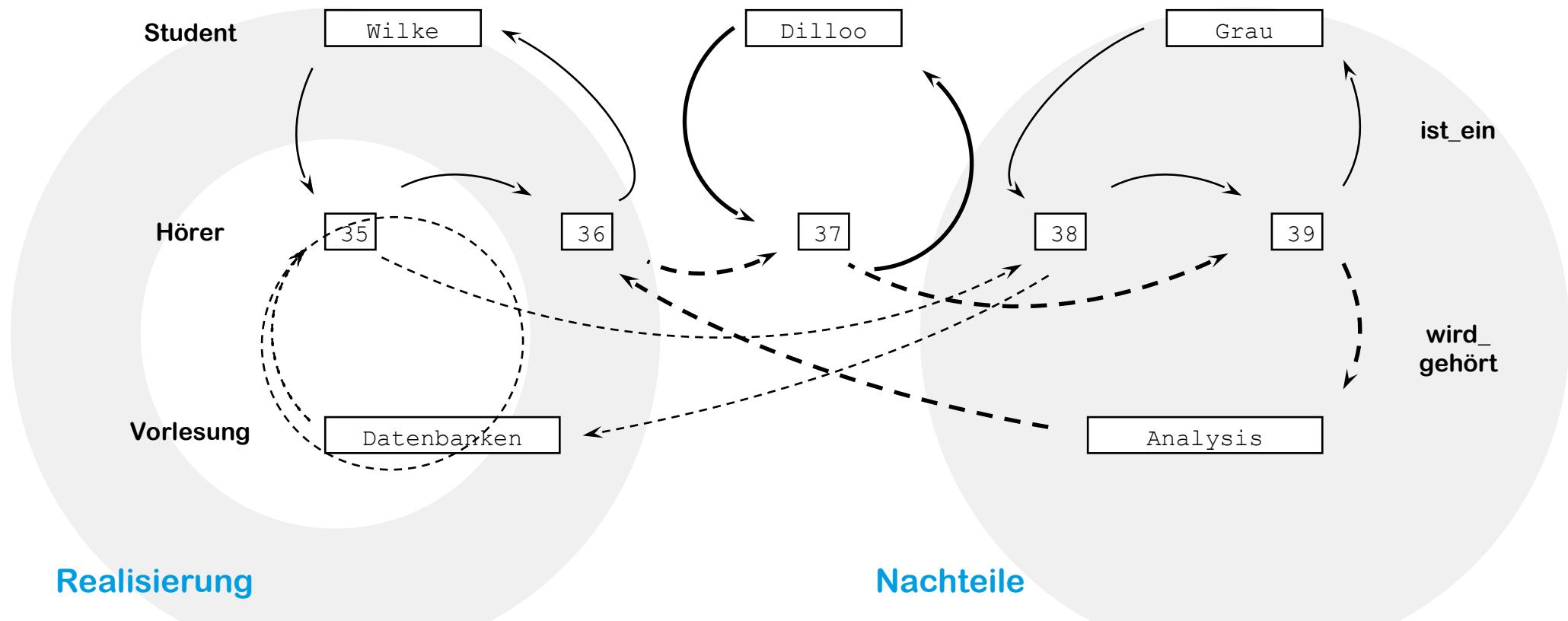
Anfragesprache

- Festlegung einer Position in Netz
- set-übergreifend navigierend





- Ausprägung -

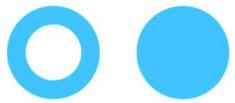


Realisierung

- ◆ Verkettung von Eigentümer-Teilnehmer-Ausprägungen der Set-Typen
- ◆ es entsteht ein Netz, da ein Record Typ zu mehreren Set-Typen gehören kann

Nachteile

- ◆ Anwender muss Speicherstruktur kennen und benutzen, um im Netz "navigieren" zu können
- ◆ Verletzung der physischen und logischen Datenunabhängigkeit



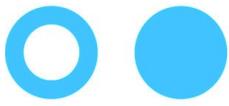
- Relationale Modell -

Inhalt

- ◆ Einführung
- ◆ Historische Datenmodelle
- ◆ **Relationale Modell**
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

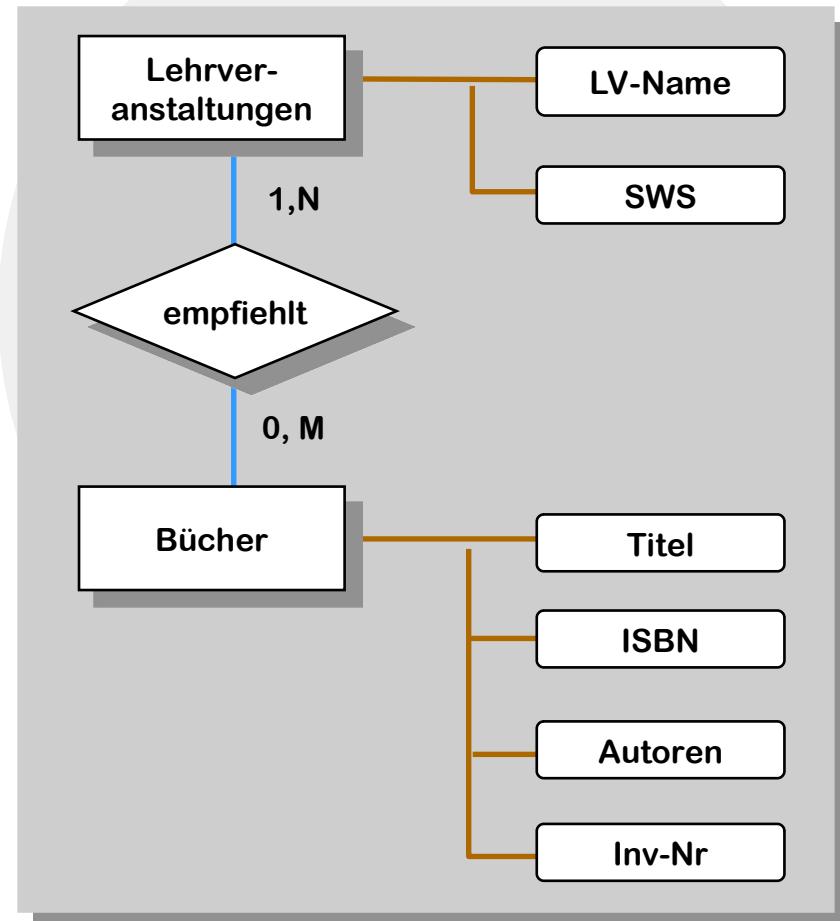
- ◆ Einführung
- ◆ Begriffe
- ◆ Eigenschaften einer Relation
- ◆ Integritätsbedingungen



- Semantisches vs. logisches Datenschema -

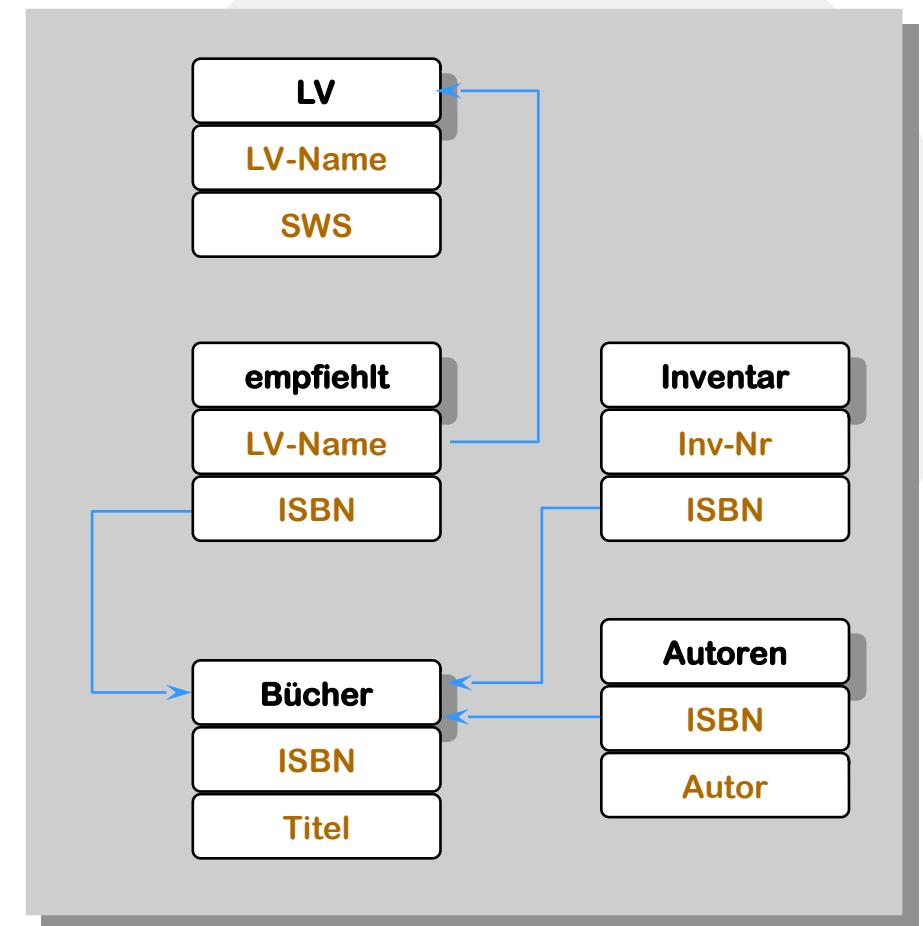
Semantisches (konzeptionelles) Schema mittels

- „Entity-Relationship“-Datenmodell

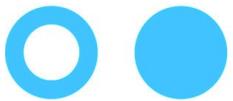


Logisches Schema mittels

- Relationen-Datenmodell



andere Bezeichnung: „relationales Datenmodell“



- Einführung -

Historie

- vorgestellt 1970 von CODD

Konzepte

- einfache Form der maschinennahen (konkreten) Datenmodellierung, ohne viele Konzepte
- einziges Konzept ist die Relation

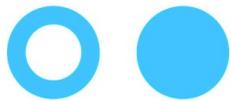
Mathematische Grundlage

- Realisierung von mächtigen Operationen auf Relationen mit Hilfe einfacher mathematischer Modelle
 - Relationenalgebra aus der Mengenlehre
 - Relationenkalkül aus der Prädikatenlogik

- geschlossenes mathematisches Modell
 - Operationen auf Relationen liefern wieder Relationen
- mathematisches Modell bildet exakte und einfache Grundlage zur Entwicklung relationaler Datenbanksprachen
 - SQL
 - QBE

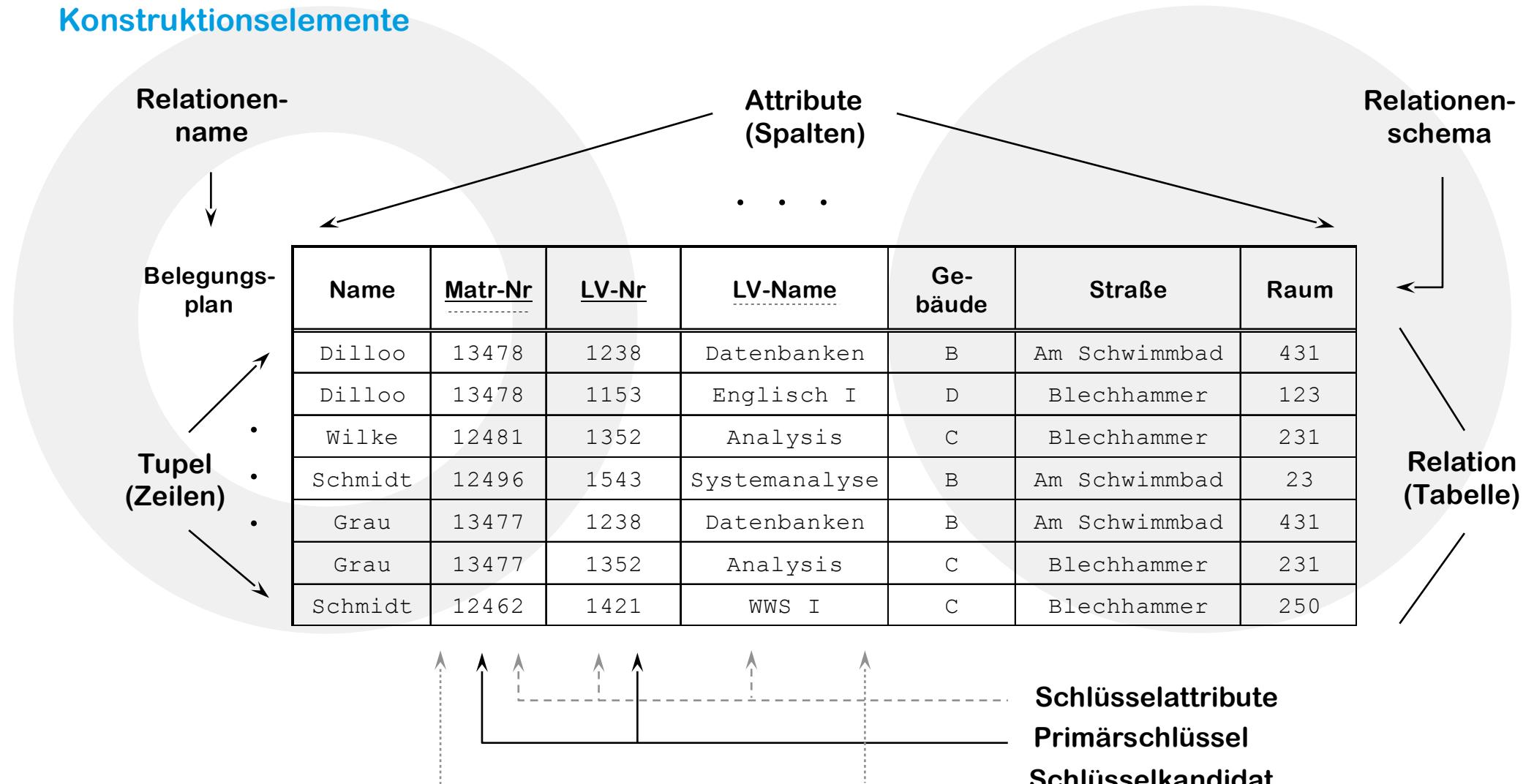
Theorie der Normalisierung

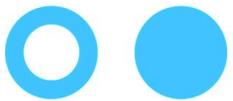
- mathematische Basis für logischen Datenbankentwurf
- erlaubt Erstellung redundanzfreier relationaler Modelle



- Eigenschaften einer Relation (I) -

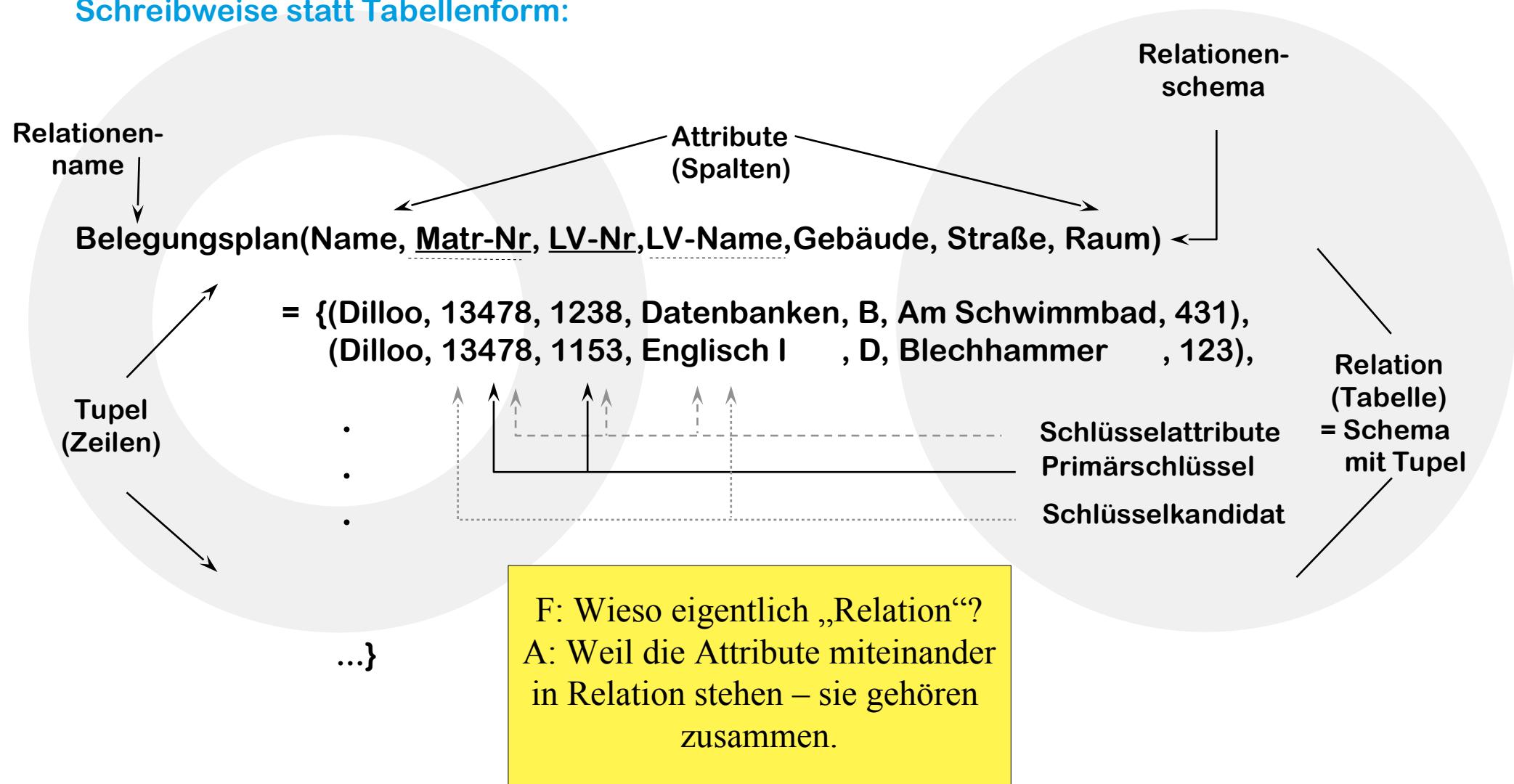
Konstruktionselemente

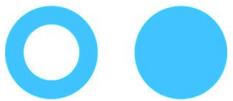




- Eigenschaften einer Relation (II) -

Schreibweise statt Tabellenform:





- Eigenschaften einer Relation (III) -

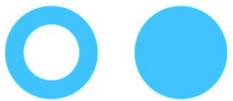
Tabelle aus Zeilen und Spalten mit Werten

- ◆ Attribut
 - Spalte einer Tabelle
 - enthält Werte desselben Wertebereichs
- ◆ Wertebereich (Domäne)
 - möglichen Werte eines Attributs
- ◆ Attributwert
 - atomares Element eines Wertebereichs
 - keine mehrwertigen Attribute
 - keine Listen oder Mengen
- ◆ Relationenschema
 - Menge von Attributen
 - Attribute sind ungeordnet

- ◆ Relation
 - Menge von Zeilen (Tupel) einer Tabelle
 - genauer: die in Beziehung bzw. Relation stehenden Werte
- ◆ Tupel
 - Zeile einer Relation
 - ein Objekt der Miniwelt (Entity)
 - Zeilen können nicht doppelt sein
 - Zeilen sind ungeordnet
- ◆ Datenbankschema
 - Menge von Relationenschemata
- ◆ Datenbank
 - Menge von Relationen

Potentielles Problem

- ◆ jeder Wert muss über Angabe von Zeile und Spalte eindeutig und atomar adressierbar sein
- ◆ Folge: keine komplexen Werte



- Integritätsbedingungen -

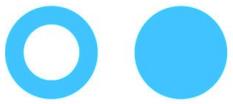
Modellinhärente Integritätsbedingungen

- ◆ **Schlüssel**
 - (minimale) Attribut (-gruppe), das ein Tupel (Zeile) der Relation eindeutig identifiziert
 - ein Schlüssel besteht mindestens aus einem Attribut, maximal aus allen Attributen des Relationenschemas
 - da kein Tupel in einer Relation doppelt vorkommen kann, besitzt jedes Relationenschema auch mindestens einen Schlüssel
- ◆ **Schlüsselattribut**
 - ein Attribut des Schlüssels
- ◆ **Primärschlüssel**
 - bei mehreren Schlüsseln (Schlüsselkandidaten), wird einer zum Primärschlüssel bestimmt

- ◆ **Primärschlüsselbedingung**
 - die Attributwerte aller Primärschlüsselelemente einer Relation müssen unterschiedlich sein
- ◆ **Fremdschlüssel**
 - Attributmenge eines Relationenschemas, die in einem anderen Relationenschema Primärschlüssel ist
- ◆ **Fremdschlüsselbedingung**
 - die Attributwerte eines Fremdschlüssels müssen in der anderen Relation als Werte des Primärschlüssels existieren

Zusätzliche Integritätsbedingungen

- ◆ Festlegung, ob bestimmte Attribute eines Tupels Werte annehmen müssen oder nicht ('NULL'-Werte)
- ◆ Festlegung disjunkter Aufteilung von Tupeln auf unterschiedliche Relationen



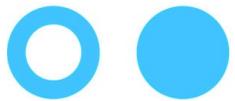
- Normalisierung -

Inhalt

- ◆ Einführung
- ◆ Historische Datenmodelle
- ◆ Relationale Modell
- ◆ **Normalisierung**
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

- ◆ Nicht normalisierte Relation
- ◆ Nachteile nicht-normalisierter Relationen
- ◆ Schritte der Normalisierung
- ◆ 1. Normalform
- ◆ 2. Normalform
- ◆ 3. Normalform
- ◆ BCNF Normalform
- ◆ Minimalität
- ◆ Referentielle Integrität



- Nicht normalisierte Relation -

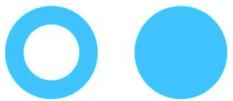
Belegungs-
plan

Wiederholungs-
gruppe

Name	<u>Matr- Nr</u>	{ Lehrveranstaltung }				
		LV- Nr	LV-Name	Ge- bäude	Straße	Raum
Dilloo	13478	1238	Datenbanken	B	Am Schwimmbad	431
		1153	Englisch I	D	Blechhammer	123
Wilke	12481	1352	Analysis	C	Blechhammer	231
Schmidt	12496	1543	Systemanalyse	B	Am Schwimmbad	23
Grau	13477	1238	Datenbanken	B	Am Schwimmbad	431
		1352	Analysis	C	Blechhammer	231
Schmidt	12462	1421	WWS I	C	Blechhammer	250

= zahlreiche Redundanzen

= Schlüssel / Schlüsselattribut



- Nachteile nicht-normalisierter Relationen -

Identifizierbarkeit der Werte

- ◆ keine eindeutigen Spalten
- ◆ Probleme bei Zugriffen auf Werte von Wiederholungsgruppen

Datenredundanz

- ◆ Speicherverschwendung
- ◆ Gefahr von Inkonsistenzen (Anomalien)
- ◆ Probleme beim Mehrbenutzerbetrieb

Anomalien können entstehen durch

- ◆ Einfügeoperation
 - eigenständige Verwaltung von Informationen der Miniwelt nicht immer möglich

Löschoperation

- wird ein Tupel gelöscht, werden u.U. auch Informationen gelöscht, die in der Miniwelt noch existieren

Modifikation

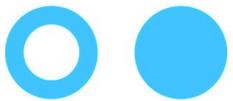
- wird ein Wert eines Tupels verändert und taucht derselbe Wert in mehreren Tupeln auf, entstehen Inkonsistenzen

Ziele der Normalisierung

- ◆ Redundanzfreiheit
- ◆ keine Anomalien bei Veränderungsoperationen
- ◆ korrektes Abbild der Realität
- ◆ Verringerung des Wartungsaufwands

Hilfsmittel

- ◆ Schritte der Normalisierung



- Schritte der Normalisierung -

Erste Normalform (1NF)

- keine Wiederholungsgruppen

Zweite Normalform (2NF)

- 1NF
- Eliminierung von Attributen, die bereits von Teilen des Schlüssels abhängig sind

Dritte Normalform (3NF)

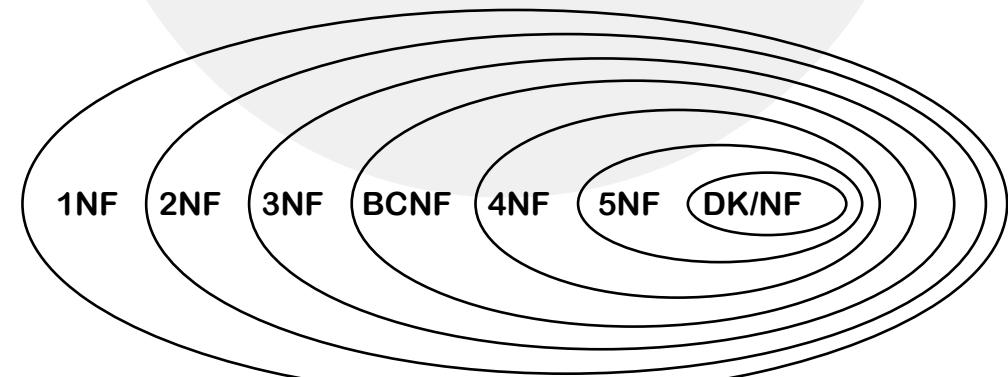
- 2NF
- Eliminierung von Attributen, die zusätzlich von nicht-Schlüsselattributen abhängig sind

Boyce Codd Normalform (BCNF)

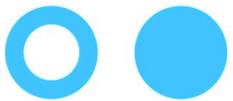
- 3NF
- zusätzlich Eliminierung von Abhängigkeiten zwischen Schlüsselattributen

Praktische Bedeutung

- 3NF oder BCNF sollte bei jedem Datenbankentwurf hergestellt werden
- die vollständige Redundanzfreiheit wird mit der Domain / Key Normalform (DK/NF) erzielt
- 4NF, 5NF und DK/NF haben eher theoretische Bedeutung, da noch keine einfachen Techniken zur Unterstützung der Entwickler gefunden wurden
- häufig wird beim physischen Entwurf wieder denormalisiert (Laufzeitgründe)



Teilmengen von Relationen



- 1. Normalform -

Ziel

- Keine Wiederholungsgruppen (zwei Dimensionen)

Ergebnis

- eindeutige Identifizierbarkeit der Werte über Zeile und Spalte
- Entstehung zusätzlicher Redundanzen

Methode

- ein Tupel pro Element einer Wiederholungsgruppe
- mindestens ein Attribut der Wiederholungsgruppe wird zu einem Schlüsselattribut

Primär-schlüssel-attribut

Belegungs-plan		Name	Matr-Nr.	LV-Nr	LV-Name	Ge-bäude	Straße	Raum
zahl-reiche neue Redun-danzen	{	Dilloo	13478	1238	Datenbanken	B	Am Schwimmbad	431
		Dilloo	13478	1153	Englisch I	D	Blechhammer	123
	{	Wilke	12481	1352	Analysis	C	Blechhammer	231
		Schmidt	12496	1543	Systemanalyse	B	Am Schwimmbad	23
	{	Grau	13477	1238	Datenbanken	B	Am Schwimmbad	431
		Grau	13477	1352	Analysis	C	Blechhammer	231
		Schmidt	12462	1421	WWS I	C	Blechhammer	250



- 2. Normalform (I) -

Ziel

- Eliminierung von nicht-Schlüsselattributen, die bereits von Teilen des Schlüssels abhängig sind (partielle Abhängigkeit vom Schlüssel)
- nur bei Relationen mit zusammengesetzten Schlüsseln

Belegungs-
plan



Name	<u>Matr-Nr.</u>	<u>LV-Nr.</u>	LV-Name	Ge- bäude	Straße	Raum
Dilloo	13478	1238	Datenbanken	B	Am Schwimmbad	431
Dilloo	13478	1153	Englisch I	D	Blechhammer	123
Wilke	12481	1352	Analysis	C	Blechhammer	231
Schmidt	12496	1543	Systemanalyse	B	Am Schwimmbad	23
Grau	13477	1238	Datenbanken	B	Am Schwimmbad	431
Grau	13477	1352	Analysis	C	Blechhammer	231
Schmidt	12462	1421	WWS I	C	Blechhammer	250

Primär-
schlüssel-
attribut

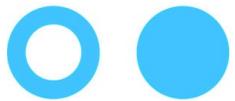
Sekundär-
schlüssel-
attribut

Methode „Analyse funktionaler Abhängigkeiten“

- Aufspalten der Relation

Ergebnis

- alle nicht-Schlüsselattribute sind vollständig vom Schlüssel abhängig
- es entstehen zusätzliche Relationen



- 2. Normalform (II) -

Relationen in 2NF

Student

Matr-Nr	Name
13478	Dilloo
12481	Wilke
12496	Schmidt
13477	Grau
12462	Schmidt

Belegungsplan

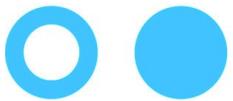
Matr-Nr	LV-Nr	LV-Name
13478	1238	Datenbanken
13478	1153	Englisch I
12481	1352	Analysis
12496	1543	Systemanalyse
13477	1238	Datenbanken
13477	1352	Analysis
12462	1421	WWS I

Lehrveranstaltung

LV-Nr	Ge-bäude	Straße	Raum
1238	B	Am Schwimmbad	431
1153	D	Blechhammer	123
1543	B	Am Schwimmbad	23
1352	C	Blechhammer	231
1421	C	Blechhammer	250

Primär-schlüssel-attribut

Sekundär-schlüssel-attribut



- 3. Normalform (I) -

Ziel

- Eliminierung von nicht-Schlüsselattributen, die zusätzlich von nicht-Schlüsselattributen abhängig sind (also transitiv vom Schlüssel abhängig sind)

Methode

- Ermittlung indirekter funktionaler Abhängigkeiten von Attributen zu Schlüsseln
- weiteres Aufspalten der Relation

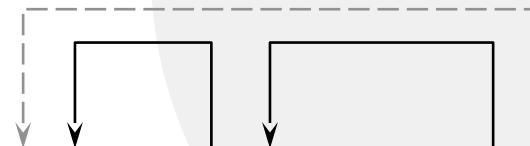
Primär-schlüssel-attribut

Sekundär-schlüssel-attribut

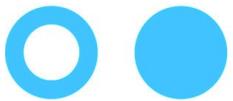
Lehr-veranstaltung

Ergebnis

- alle Attribute sind ausschließlich vom Schlüssel abhängig
- weniger Redundanzen
- zusätzliche Relationen

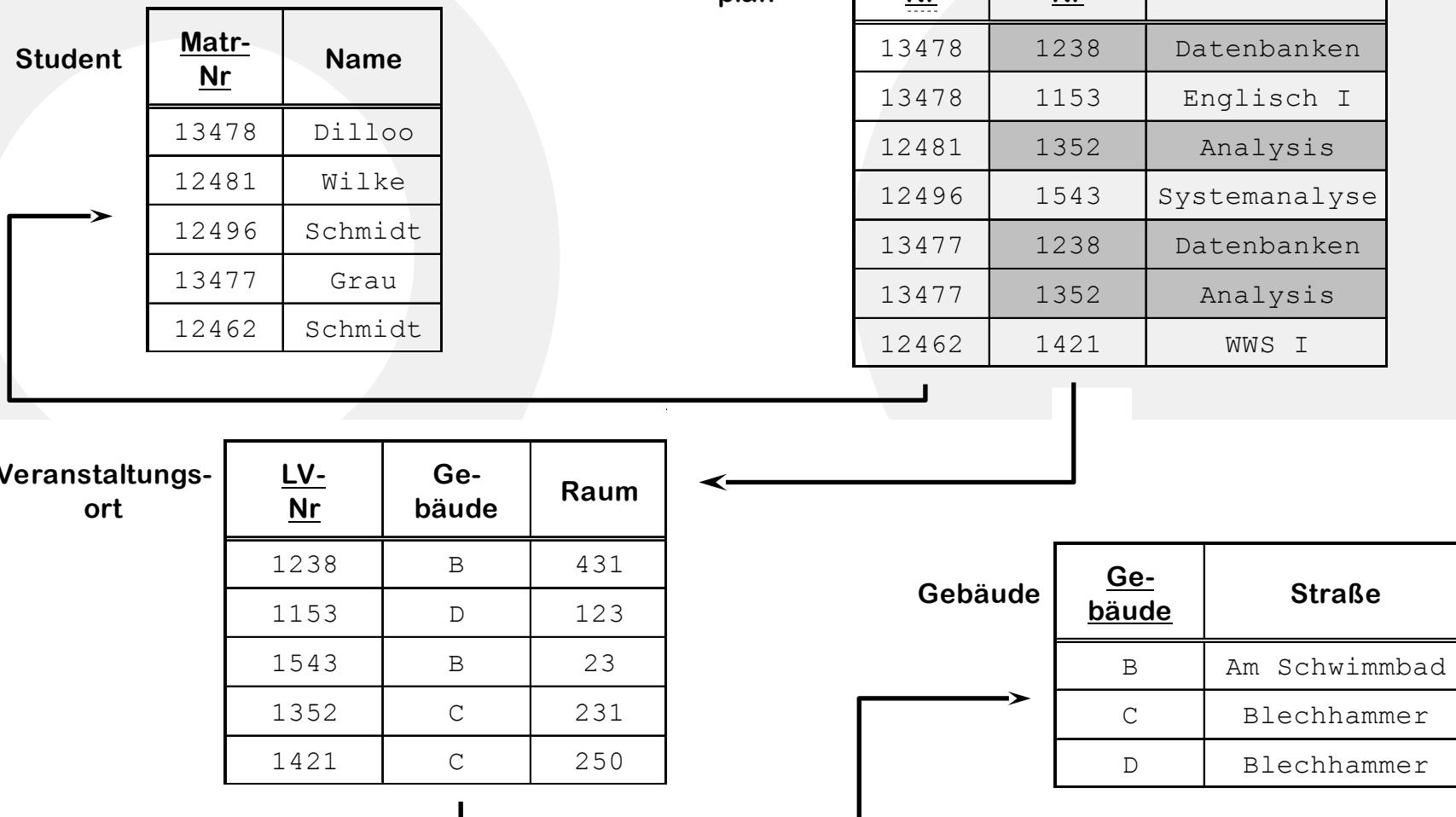


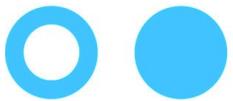
LV-Nr	Ge-bäude	Straße	Raum
1238	B	Am Schwimmbad	431
1153	D	Blechhammer	123
1543	B	Am Schwimmbad	23
1352	C	Blechhammer	231
1421	C	Blechhammer	250



- 3. Normalform (II) -

Relationen in 3NF





- BCNF Normalform (I) -

Ziel

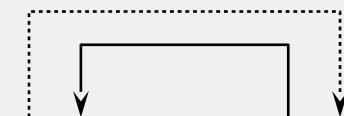
- Eliminierung von Schlüsselattributen,
 - die nicht zum Primärschlüssel gehören
 - und bereits von Teilen des Primärschlüssels abhängig sind

Methode

- Ermittlung weiterer funktionaler Abhängigkeiten zwischen den Schlüsselattributen
- weiteres Aufspalten der Relation

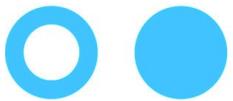
Ergebnis

- keine (offensichtlichen) Abhängigkeiten mehr zwischen Teilen von Schlüsseln
- nahezu Redundanzfreiheit
- zusätzliche Relationen



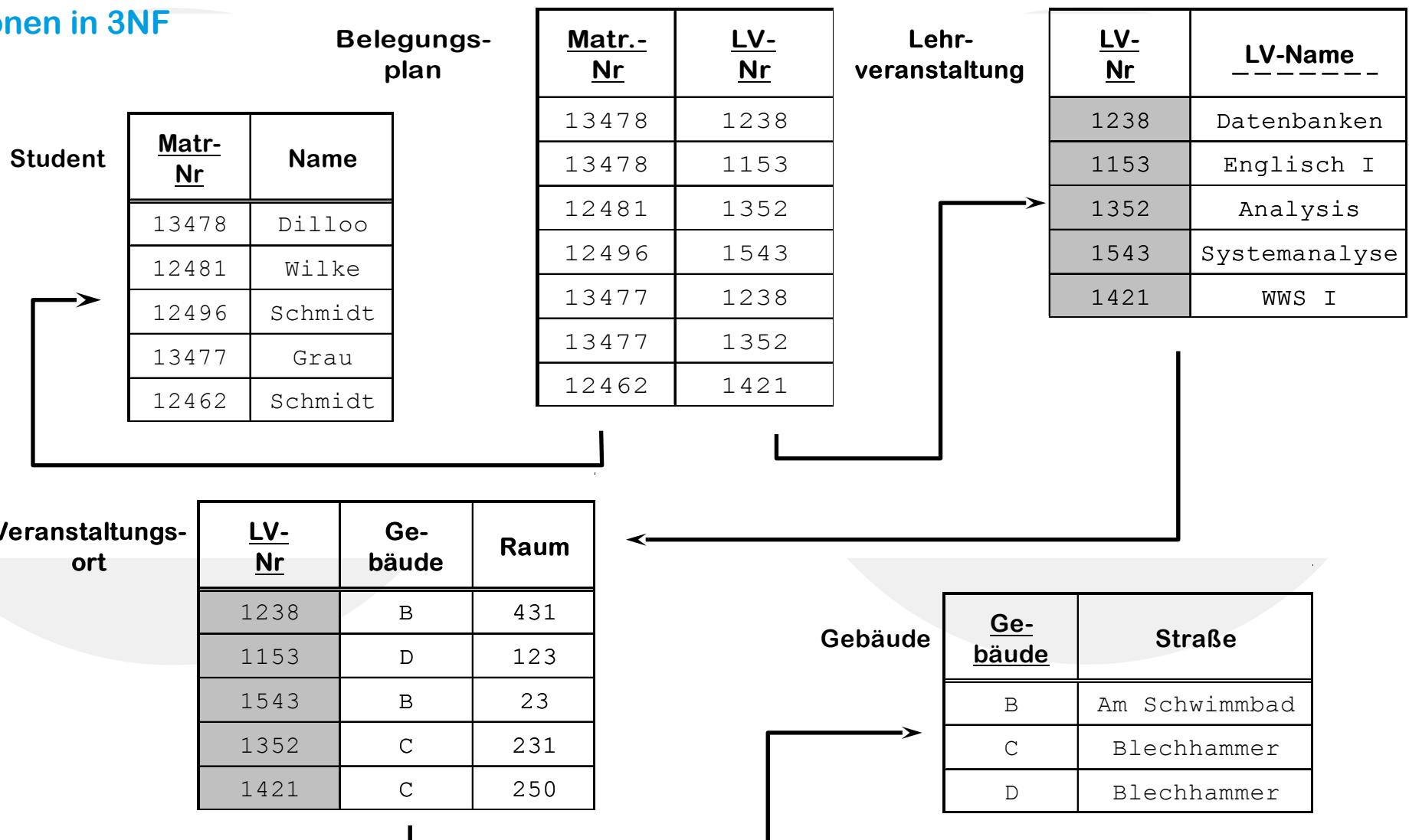
Belegungs-
plan

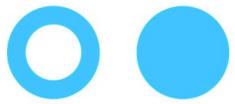
Matr.-Nr.	LV-Nr.	LV-Name
13478	1238	Datenbanken
13478	1153	Englisch I
12481	1352	Analysis
12496	1543	Systemanalyse
13477	1238	Datenbanken
13477	1352	Analysis
12462	1421	WWS I



- BCNF Normalform (II) -

Relationen in 3NF





- Minimalität -

Minimale Anzahl
Relationen in BCNF

Student



Belegungs-
plan

Matr.-Nr	Name
13478	Dilloo
12481	Wilke
12496	Schmidt
13477	Grau
12462	Schmidt

Matr.-Nr	LV-Nr
13478	1238
13478	1153
12481	1352
12496	1543
13477	1238
13477	1352
12462	1421

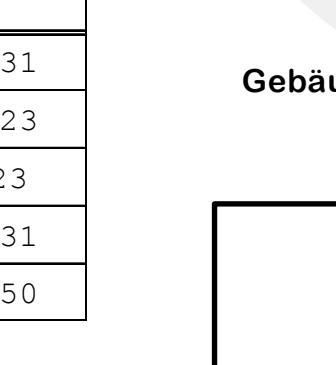
Veranstaltung

Primär-
schlüssel-
attribut

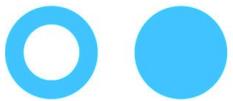
Sekundär-
schlüssel-
attribut

LV-Nr	LV-Name	Ge- bäude	Raum
1238	Datenbanken	B	431
1153	Englisch I	D	123
1352	Analysis	B	23
1543	Systemanalyse	C	231
1421	WWS I	C	250

Gebäude



Ge- bäude	Straße
B	Am Schwimmbad
C	Blechhammer
D	Blechhammer



- Referentielle Integrität -

Relationen in BCNF mit max-Kardinalitäten

Student	Matr.-Nr	Name
N	13478	Dilloo
	12481	Wilke
	12496	Schmidt
	13477	Grau
	12462	Schmidt

Belegungs-plan

Matr.-Nr	LV-Nr
13478	1238
13478	1153
12481	1352
12496	1543
13477	1238
13477	1352
12462	1421

Lehr-veranstaltung

LV-Nr	LV-Name
1238	Datenbanken
1153	Engkish I
1352	Analysis
1543	Systemanalyse
1421	WWS I

1 (N) ???

Veranstaltungs-ort

LV-Nr	Ge-bäude	Raum
1238	B	431
1153	D	123
1543	B	23
1352	C	231
1421	C	250

Primär-schlüssel-attribut

Sekundär-schlüssel-attribut

N ???

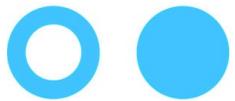
Primärschlüssel
Fremdschlüssel
Schlüsselkandidat

1

Gebäude

Ge-bäude	Straße
B	Am Schwimmbad
C	Blechhammer
D	Blechhammer

N



- Abbildung eines ER- auf relationales Schema -

Inhalt

- ◆ Einführung
- ◆ Historische Datenmodelle
- ◆ Relationale Modell
- ◆ Normalisierung
- ◆ Abbildung eines ER- auf relationales Schema

Überblick

- ◆ Kriterien
- ◆ 1:1-Beziehung
- ◆ N:M-Beziehung
- ◆ 1:N-Beziehung
- ◆ Muss-Beziehung
- ◆ Generalisierung / Spezialisierung
- ◆ Zusammenfassung