To start….

Please, close your laptops

and just

Prof. DI Dr. Erich Gams

Relationale Entwurfstheorie
Normalisierung

Integrität, Anomalien und Normalformen

informationssysteme htl-wels

Übersicht ☛ Was lernen wir?

- Kurze Wiederholung
- Datenintegrität
- Anomalien
- Normalformen
- Übungen

Begriff: Datenintegrität

- Unter dem Begriff Integrität oder Konsistenz (engl. integrity, consistency) versteht man die Widerspruchsfreiheit von Datenbeständen.
- Eine Datenbank ist integer oder konsistent, wenn die gespeicherten Daten
  - fehlerfrei erfasst sind und
  - den gewünschten Informationsgehalt korrekt wiedergeben.
- Die Datenintegrität ist verletzt, 
  - Mehrdeutigkeiten oder widersprüchliche Sachverhalte treten auf.

Schlüssel-Integritätsbedingung

- ⇨ Relationen sind Mengen von Tupeln, die allein durch ihre Werte unterschieden werden.
- ⇨ Der Begriff Menge impliziert Eindeutigkeit der Elemente,
  - ⇨ d. h. es kann in einer Menge nicht zwei Elemente geben, die die gleichen Werte besitzen.
- ⇨ Tupel müssen folglich eindeutig identifizierbar sein.
  -> Somit muss also auch jeder Schlüssel eindeutig sein.

Gegenstands-Integritätsbedingung

- Die Gegenstands-Integritätsbedingung folgt direkt aus der Schlüssel-Integritätsbedingung und besagt, dass **kein Primärschlüsselwert NULL** (=kein Wert) **sein darf**.
  - NULL-Werte für Schlüsselattribute
  - -> mehrere Tupel NULL als Schlüsselwert
  - -> Tupel nicht mehr eindeutig identifizierbar

Rechn#	Datum	Name	Vorname
NULL	12.05.2020	Bach	Bodo
NULL	13.05.2020	Sophia	Weiß

Referenzielle Integritätsbedingung

- Referenzielle Integritätsbedingungen verlangen, dass **aktuelle Fremdschlüsselwerte sich immer nur auf Primärschlüsselwerte von existierenden Tupeln beziehen.**

Mitarbeiter

ID	Nachname	Abteilung
1	Müller	A1
2	Meier	A3
3	Tobler	A2

Abteilung

Abt_Nr	Professor
A1	Informatik
A2	Marketing
A3	Finance

Integritätsgefährdende Operationen - Anomalien

- Wir unterscheiden und erläutern drei Arten von Operationen, die die Integrität im besprochenen Sinne gefährden können:
  - Einfügen von Tupeln (Einfügeanomalie)
  - Löschen von Tupeln (Löschanomalie)
  - Ändern von Attributwerten eines Tupels (Updateanomalie)

Anomalien Beispiel

⇨ Änderungs (Update)-Anomalien

Rechn#	Datum	Name	Vorname	Str	Nr	Ort	Artikel	Anzahl	Preis
2334	12.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Bleistift	10	1 Euro
2335	13.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Papier A4	5	2 Euro
2336	20.05.2020	Sophia	Weiß	Goethekreuzung	23	4010 Linz	Drucker	1	199 Euro

- ⇨ Kunden mit mehreren Bestellungen -> Daten mehrfach erfasst und gespeichert.
- ⇨ Kundendaten für denselben Kunden müssen immer übereinstimmen.
- ⇨ Adressänderung muss an mehreren (hier 2) Stellen korrekt geändert werden -> fehleranfällig

Anomalien Beispiel

⇨ Einfüge (Insert)-Anomalien

Rechn#	Datum	Name	Vorname	Str	Nr	Ort	Artikel	Anzahl	Preis
2334	12.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Bleistift	10	1 Euro
2335	13.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Papier A4	5	2 Euro
2336	20.05.2020	Sophia	Weiß	Goethekreuzung	23	4010 Linz			

⇨ Eine Person kann in dieser Datenbank nicht eingefügt werden, wenn sie noch keine Bestellung gemacht hat, d. h. die Datenbank ist nur für Bestellungen zu gebrauchen und z. B. nicht gleichzeitig als Kontaktdatei

Anomalien Beispiel

⇨ Lösch (Delete)-Anomalien

Rechn#	Datum	Name	Vorname	Str	Nr	Ort	Artikel	Anzahl	Preis
2334	12.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Bleistift	10	1 Euro
2335	13.05.2020	Bach	Bodo	Bahnhofstr	3	4600 Wels	Papier A4	5	2 Euro

⇨ Werden alte Bestelldaten gelöscht -> geht Information über den Kunden verloren. Die Adresse könnte aber bspw. noch benötigt werden.

Lösung: Normalisierung

- Aufteilung einer Tabelle in mehrere Tabellen, damit möglichst keine unerwünschten Nebeneffekte auftreten.
- Diese Aufteilung erfolgt mit dem Verfahren der **Normalisierung**.

- ***„Unter Normalisierung eines relationalen Datenschemas versteht man die schrittweise Zerlegung von Relationen, um Redundanzen innerhalb des Datenschemas zu vermeiden“ (Wikipedia)***

Ziele der Normalisierung

- Eine Relation ist:
  - Sie ist redundanzfrei.
  - Sie verursacht keine Probleme bei der Datenpflege.
  - Sie beschreibt einen Ausschnitt aus der Realität angemessen und richtig.

Abhängigkeiten

- Um die Umwandlung der Relationen in die drei Normalformen zu verstehen, müssen wir zuerst das Konzept der Abhängigkeiten zwischen Attributen dieser Relationen einführen.

Funktionale Abhängigkeit

- Attribut **B ist von** Attribut **A funktional abhängig**, wenn zu jedem Wert von A höchstens ein Wert von B auftreten kann.
  A --> B

- Beispiel:

ID	Name
S1	Meier
S2	Weber

- Das Attribut Name ist funktional abhängig vom Attribut ID (ID --> Name).

Funktionale Abhängigkeit

- Erklärung:

- Ein Attribut B heißt **funktional abhängig** vom Attribut A, falls zu einem Wert von Attribut A höchstens ein Wert von B gehört.

- So sind z.B.: Name und Vorname einer Person funktional abhängig von der Personalnummer (Primärschlüssel) dieser Person.

(Primär-)Identifikationsschlüssel

- Ein Attribut A für das gilt: Jedes Attribut ist von A funktional abhängig; kein Attribut von A ist von den übrigen A-Attributen funktional abhängig.
  A --> G
- Beispiel:

ID	Name	Vorname
S1	Meier	Hans
S2	Weber	Ueli

- Das Attribut ID ist Primärschlüssel.

Volle funktionale Abhängigkeit

- ⇨ A sei der Primärschlüssel, B Attribut;
- ⇨ B ist genau dann von A voll funktional abhängig, wenn B von A funktional abhängig ist, aber nicht bereits von Teilen von A.
  - ⇨ A ==> B
- ⇨ Beispiel:

IDStudent	Name	IDProfessor	Note
S1	Meier	P2	5
S2	Weber	P1	6

- ⇨ Das Attribut „Note“ ist voll funktional abhängig von den Attributen „IDStudent“ und „IDProfessor“ („IDSt, IDProf ==> Note“). „Name“ aber nicht.

Volle funktionale Abhängigkeit

- Umgekehrt formuliert heißt dies: Eine Tabelle ist noch nicht in zweiter Normalform, wenn sie einen zusammengesetzten Primärschlüssel hat und ein Nichtschlüssel-Attribut nicht vom ganzen Primärschlüssel, sondern nur von einem Teilschlüssel abhängt
- Hinweis:
  - Besteht der Schlüssel A nur aus einem Attribut und ist B funktional abhängig von A, so ist B bereits voll funktional abhängig.

Transitive Abhängigkeit

- A sei der Primärschlüssel, B und C sind weitere Attribute, alle untereinander verschieden/disjunkt;
- C ist transitiv abhängig von A wenn gilt:
  A --> B ; B --> C ; B -/-> A
- Beispiel:

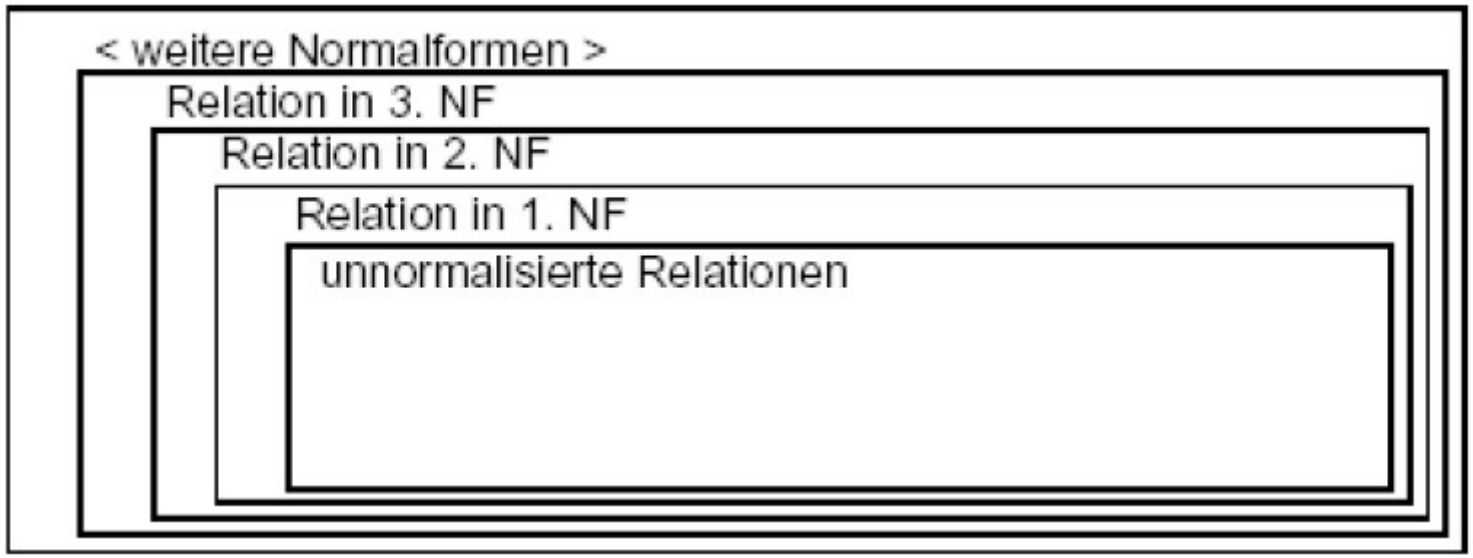
SchülerNr	Name	Klasse	KV
1	Förster	1BHIT	Laage
2	Weißmüller	2BHIT	Game
3	Urner	4AHIT	Lois
4	Lehmann	3AHIT	Helt
5	Freytag	2BHIT	Game

- Die funktionale Abhängigkeit bezüglich *Klasse --> KV* ist eine transitive Abhängigkeit, da *Klasse* nicht Primärschlüssel der Relation ist.

Transitive Abhängigkeit

- Erklärung:
- Das Attribut C heißt **transitiv abhängig** von A, falls es ein Nicht-Schlüssel-Attribut B gibt, das funktional abhängig ist von A und von dem C funktional abhängt.

Normalformen Übersicht



Jede Normalform enthält implizit die vorgehende Normalform (die 3. Normalform enthält die zweite und damit die erste Normalform). Um auf eine Normalform zu kommen, müssen nicht zwangsweise die vorgehenden Normalformen durchlaufen werden (d.h. man kann z.B. mit etwas Übung direkt auf die 3. NF kommen).

1. Normalform

- Ein Relationenschema befindet sich in der 1. Normalform, wenn alle seine Attribute einfach und einwertig (atomar) sind.
- Zur Verwaltung der Studenten sei folgende Relation gegeben:

Studenten

Vorname	Nachname	Informatikkenntnisse
Thomas	Müller	Java, C++, PHP
Ursula	Meier	PHP, Java
Igor	Müller	C++, Java

1. Normalform

Studenten

Vorname	Nachname	Informatikkenntnisse
Thomas	Müller	Java, C++, PHP
Ursula	Meier	PHP, Java
Igor	Müller	C++, Java

Ausgangslage

Resultat nach Normalisierung

Studenten

Vorname	Nachname	Informatikkenntnisse
Thomas	Müller	C++
Thomas	Müller	PHP
Thomas	Müller	Java
Ursula	Meier	Java
Ursula	Meier	PHP
Igor	Müller	Java
Igor	Müller	C++

Beispiel 1. Normalform

2. Normalform

- Ein System von Tabellen ist dann in der **zweiten Normalform** (**NF2**), wenn die Tabellen in der ersten Normalform sind und wenn zusätzlich alle Nichtschlüssel-Attribute voll funktional vom Primärschlüssel abhängig sind.

- Umgekehrt formuliert heißt dies:
- Eine Tabelle ist noch nicht in zweiter Normalform, wenn sie einen zusammengesetzten Primärschlüssel hat und ein Nichtschlüssel-Attribut nicht vom ganzen Primärschlüssel, sondern nur von einem Teilschlüssel abhängt.

2. Normalform

- Hinweis: Die 2.Normalform kann nur verletzt werden, wenn der Primärschlüssel aus mehr als einem Attribut zusammengesetzt ist.

- Auflösung in 2.NF:
- Felder, die nur von einem Schlüsselteil abhängen, müssen separat modelliert werden.

2 NF Beispiel

Studenten

IDSt	Nachname	IDProf	Professor	Note
1	Müller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	6

- Die Attribute *IDSt* und *IDProf* bilden den Primärschlüssel.
- Alle Attribute sind einfach und einwertig, d.h. die obere Tabelle ist in der 1. Normalform.
- Wenn jedoch der Student 1 von der Schule abgeht und gelöscht wird, gehen auch alle Informationen über den Professor Schmid verloren.

2. Normalform

- Zudem ist bekannt, dass folgende funktionale Abhängigkeiten existieren:

- „IDProf --> ProfessorNachname“.
  - Das Attribut „*ProfessorNachname*“ ist funktional abhängig vom Attribut „IDProf“ ()

- „IDSt --> StudentNachname“
  - Das Attribut „StudentNachname“ ist funktional abhängig vom Attribut „IDSt“
  -

- „IDSt, IDProf ==> Note“
  - Das Attribut „Note“ ist voll funktional abhängig von den Attributen „IDSt“ und „IDProf“

2.Normalform

Studenten

IDSt	Nachname	IDProf	Professor	Note
1	Müller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	6

Ausgangslage

Resultat nach Normalisierung

Studenten

ID	Nachname
1	Müller
2	Meier
3	Tobler

Professoren

IDProf	Professor
1	Bernasconi
2	Borner
3	Schmid

Noten

IDST	IDProf	Note
1	3	5
2	2	4
3	1	6

Beispiel 2. Normalform

2.Beispiel

Rechnungs#	Artikel#	Artikelname	Anzahl	Preis
1	4711	Kanu	1	799
1	4712	SUP	2	350
1	4713	Paddel	2	49
2	4712	SUP	3	350

2.Beispiel

- ⇨ Artikelname ist nicht abhängig von Rechnungs# UND Artikel#, sondern NUR von Artikel#.
- ⇨ Auflösung der „teilweisen“ Relation durch Einführung einer weiteren Tabelle. (= Artikel)

Rechnungs#	Artikel#	Anzahl
1	4711	1
1	4712	2
1	4713	2
2	4712	3

Artikel#	Artikelname	Preis
4711	Kanu	799
4712	SUP	350
4713	Paddel	49
4712	SUP	350

3. Normalform

- Ein Relationenschema befindet sich in der 3. Normalform,
  - wenn es in der 2. Normalform ist und
  - kein Attribut, das nicht zum Primärschlüssel gehört, von diesem transitiv abhängt.
- Erklärung:
- Alle Nichtschlüssel-Attribute sind voneinander unabhängig
- Auflösung:
- Die transitiv abhängigen Datenfelder werden in weitere Tabellen ausgelagert

3. Normalform

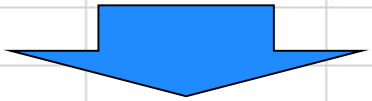
- Zur Verwaltung der Bankverbindung von Lieferanten sei folgende Relation gegeben:

SchülerNr	Name	Klasse	KV
1	Förster	1BHIT	Laage
2	Weißmüller	2BHIT	Game
3	Urner	4AHIT	Lois
4	Lehmann	3AHIT	Helt
5	Freytag	2BHIT	Game

- Das Attribut *SchülerNr* ist Primärschlüssel.
- Folgende funktionale Abhängigkeiten existieren:
  - *Name, Klasse, KV* sind funktional abhängig von *SchülerNr*
    - *SchülerNr --> Name, Klasse, KV*
  - *KV* ist funktional abhängig von *Klasse*
    - *Klasse --> KV*

3.Normalform

SchülerNr	Name	Klasse	KV
1	Förster	1BHIT	Laage
2	Weißmüller	2BHIT	Game
3	Urner	4AHIT	Lois
4	Lehmann	3AHIT	Helt
5	Freytag	2BHIT	Game



SchülerNr	Name	Klasse
1	Förster	1BHIT
2	Weißmüller	2BHIT
3	Urner	4AHIT
4	Lehmann	3AHIT
5	Freytag	2BHIT

Klasse	KV
1BHIT	Laage
2BHIT	Game
4AHIT	Lois
3AHIT	Helt

Beispiel

- Ist folgende Tabelle in 3.NF?

Reise

Rechnungsnummer	Datum	Name	Vorname	Straße	PLZ	Ort

Lösung

Reise

Rechnungsnummer	Datum	Personalnummer

Personal

Personalnummer	Name	Vorname	Straße	PLZ

PLZ

PLZ	Ort

Fragen?