Normalisierung – Regeln zum Minimieren von Redundanzen

Durch Normalisierung und strenge Regeln soll eine korrekte, relationale Datenbank aufgebaut werden. Wichtig ist es, dass Redundanzen vermieden werden, da diese zu Inkonsistenzen- oder zu falschen Daten führen können und werden. Inkonsistenzen sind Widersprüchlichkeiten der Daten.

Beispiel:

Mitarbeiter pflegen die Daten der Kundendatenbank. Dazu können bei jedem Kunden die PLZ und der Ort unabhängig voneinander eingetragen werden.

Der erste Mitarbeiter trägt als PLZ „3550“ und als Ort „Langnau im Emmental“ ein.

Der zweite Mitarbeiter als PLZ „3550“ und als Ort „Langnau i/E“ ein. Bereits liegt eine Inkonsistenz vor.

Der nächste Mitarbeiter trägt für den nächsten Kunden als PLZ wieder „3550“ ein und als Ort dann „Langnau“.

Am Abend kommt der Chef und lässt sich einer Statistik ausgeben, wie viele Kunden aus „Langnau“ eingetragen wurden – er bekommt nur einen. Hätte er nach der PLZ „3550“ gesucht, hätte er 3 Kunden angezeigt bekommen.

Dieses Beispiel zeigt, wie schnell eine Datenbank zu unbeständigen, inkonsistenten Daten kommen kann wenn dies nicht verhindert wird.

Normalformen

Es gibt sechs verschiedene Schritte (Normalformen) welche zum Bereinigen von Datenmodellen zum Einsatz kommen. In der Praxis werden jedoch nur die ersten drei Normalformen umgesetzt.

- 1. Normalform (1NF)
- 2. Normalform (2NF)
- 3. Normalform (3NF)
- Boyce-Codd Normalform (BCNF)
- 4. Normalform (4NF)
- 5. Normalform (5NF)

Je höher die Normalform, desto strenger sind die Regeln zum Eliminieren von Redundanzen.

Die Voraussetzung jeder Normalform ist, dass die vorherige Normalform gegeben ist. Die NF‘s bauen aufeinander auf.

- 2NF bedeutet, dass 1NF erfüllt ist und ...
- 3NF bedeutet, dass 2NF erfüllt ist und ...
- etc. etc.

Zweck der Normalisierung

Durch Anwendung der Normalisierung soll die Integrität der Daten sichergestellt werden.

- Redundanzen unterbinden
- Inkonsistenzen vermeiden

Die Wartung der Daten wird vereinfacht, die Programmierung allerdings aufwendiger. Es dreht sich schlussendlich aber alles um die Daten. Die Daten stehen im Zentrum. Die Daten bleiben lange persistent, sie definieren ein Unternehmen. Applikationen hingegen sind einem gewissen Trend unterworfen und ändern sich kontinuierlich. Darum ist es sehr wichtig den Daten den Stellenwert zu verleihen und entsprechend sorgsam damit umzugehen.

Erste Normalform (1NF)

Kernaussage

*Jedes Attribut darf nur **gleichartigen Inhalt** enthalten.*

Oder:

Jedes Attribut muss einen atomaren Wertbereich haben. Die Relation (Tabelle) muss frei von Wiederholungsgruppen sein.

Beschreibung

Atomarer Wertbereich:

Jedes Attribut muss in seine atomaren Teile zerlegt werden.

Beispiel:

Verletzt 1NF
Adresse
Bernstr. 52 3014 Bern

1NF konform			
Strasse	Nummer	PLZ	Ort
Bernstr.	52	3014	Bern

Frei von Wiederholungsgruppen:

Frei von Wiederholungsgruppen bedeutet, dass gleichartige Wertelisten nicht im gleichen Attribut untergebracht werden dürfen.

Beispiel:
In einem Attribut „Telefon“ darf nur eine Telefonnummer stehen, nicht eine Liste mit mehreren Telefonnummern.

Vor der Normalisierung in 1NF

CD_SONG

CD_ID	*ALBUM*	*JAHR*	*SONGLIST*
1001	Jeff Beck – Guitar Shop	1989	1. Guitar Shop, 2. Savoy, 3. Behind the Veil
1002	Monster Magnet – 4-Way Diablo	2007	1. Wall of Fire, 2. You‘re Alive
1003	Larry Carlton – Sleepwalk	1982	1. Last Nite
1004	Monster Magnet – God Says No	2000	1. Melt

- Das Attribut „ALBUM“ enthält Interpret und Albumtitel, ist also nicht atomar.
- Das Attribut „SONGLIST“ enthält ...
  - ...die Tracknummer und den Songtitel, ist also nicht atomar
  - ...eine Wiederholungsgruppe (mehrere Songs)

Nach der Normalisierung in 1NF

CD_SONG

CD_ID	*INTERPRET*	*ALBUM*	*JAHR*	*TRACK*	*SONG*
1001	Jeff Beck	Guitar Shop	1989	1	Guitar Shop
1001	Jeff Beck	Guitar Shop	1989	2	Savoy
1001	Jeff Beck	Guitar Shop	1989	3	Behind the Veil
1002	Monster Magnet	4-Way Diablo	2007	1	Wall of Fire
1002	Monster Magnet	4-Way Diablo	2007	2	You‘re Alive
1003	Larry Carlton	Sleepwalk	1982	1	Last Nite
1004	Monster Magnet	God Says No	2000	1	Melt

- Die Werteliste „ALBUM“ wurde in die zwei atomaren Attribute „INTERPRET“ und „ALBUM“ aufgeteilt
- Die Werteliste „SONGLIST“ wurde in die atomaren Attribute „TRACK“ und „SONG“ aufgeteilt
- Die Wiederholungsgruppe „SONGLIST“ wurde in mehrere Datensätze aufgeteilt

Jetzt ist jedes Attribut atomar und ist keine Werteliste mehr.

Jeder Datensatz kann eindeutig durch einen Primärschlüssel, zusammengesetzt durch „CD_ID“ und „TRACK“, identifiziert werden. Eine eindeutige Identifizierung ist nicht explizit Voraussetzung für 1NF, es ist jedoch so betrachdet das Resultat von 1NF.

--> Die erste Normalform ist gegeben.

Zweite Normalform (2NF)

Kernaussage

Eine Relation ist in der zweiten Normalform wenn die erste Normalform erfüllt ist und jeder Datensatz nur genau einen Sachverhalt abbildet. Liegen in einer Relation Daten vor, die mehr als einen Sachverhalt abbilden, müssen diese in mehrere Relationen aufgeteilt werden.

Anders ausgedrückt:

Eine Relation ist in der zweiten Normalform wenn die erste Normalform vorliegt und kein Nichtprimärschlüsselattribut funktional von einer Teilmenge eines PK Attributs abhängt.

Oder*:*

Jedes Nicht- PK Attribut ist jeweils vom ganzen Primary Key abhängig, nicht nur von einem Teil des Schlüssels. Wichtig ist hierbei, dass die Nichtschlüsselattribute wirklich von allen PK Attributen abhängen.

Dadurch werden Redundanzen und damit die Gefahr von Inkonsistenzen reduziert. Nur noch logisch/sachlich zusammengehörige Informationen sind in einer Relation. Dadurch fällt auch das Verständnis der Datenstrukturen leichter.

Somit ergibt sich, dass Relationen in der 1NF, deren Primärschlüssel nicht zusammengesetzt sind, sondern lediglich aus einem einzelnen Attribut bestehen, automatisch die 2NF erfüllen.

Vor der Normalisierung in 2NF

CD_SONG

CD_ID	*INTERPRET*	*ALBUM*	*JAHR*	*TRACK*	*SONG*
1001	Jeff Beck	Guitar Shop	1989	1	Guitar Shop
1001	Jeff Beck	Guitar Shop	1989	2	Savoy
1001	Jeff Beck	Guitar Shop	1989	3	Behind the Veil
1002	Monster Magnet	4-Way Diablo	2007	1	Wall of Fire
1002	Monster Magnet	4-Way Diablo	2007	2	You‘re Alive
1003	Larry Carlton	Sleepwalk	1982	1	Last Nite
1004	Monster Magnet	God Says No	2000	1	Melt

- Die Relation CD_SONG bildet mehrere Sachverhalte ab:
  - Album
  - Song
- Der Primärschlüssel der Tabelle CD_SONG ist aus 2 Attributen zsammengesetzt
- Die Attribute INTERPRET, ALBUM und JAHR sind vom Teil-Primärschlüssel CD_ID abhängig, aber nicht vom Teil-Primärschlüssel TRACK.
  - Dies verletzt die 2. Normalform, da nicht-PK Attribute nicht nur von einem Teil des Schlüssels (CD_ID) abhängen dürfen.

Problematik

Deutlich ist im Beispiel auch zu sehen, dass die Daten mehrfach (redundant) vorhanden sind. Dies wird früher oder später zu Inkonsistenzen in den Daten führen.

Nach der Normalisierung in 2NF

CD

CD_ID	*INTERPRET*	*ALBUM*	*JAHR*
1001	Jeff Beck	Guitar Shop	1989
1002	Monster Magnet	4-Way Diablo	2007
1003	Larry Carlton	Sleepwalk	1982
1004	Monster Magnet	God Says No	2000

SONG

CD_ID	*TRACK*	*SONG*
1001	1	Guitar Shop
1001	2	Savoy
1001	3	Behind the Veil
1002	1	Wall of Fire
1002	2	You‘re Alive
1003	1	Last Nite
1004	1	Melt

Die Daten in der Tabelle CD_SONG werden in zwei Tabellen aufgeteilt: CD und SONG.

Die Tabelle CD enthält nur noch Felder, die voll funktional von CD_ID abhängen, hat also CD_ID als Primärschlüssel. Da keine weiteren Schlüsselkandidaten existieren, ist die Tabelle damit in der 2. Normalform.

Die Tabelle SONG enthält nur noch Felder, die voll funktional von CD_ID und TRACK abhängen, liegt also auch in der 2. Normalform vor.

Mit Hilfe dieser Aufteilung sind auch die Redundanzen der Daten beseitigt worden.

Das Attribut CD_ID in der Tabelle SONG ist ein Fremdschlüssel (Foreign Key), welcher auf den Primary Key der Tabelle CD verweist.

Dritte Normalform (3NF)

Kernaussage

Eine Relation ist in der dritten Normalform wenn die zweite Normalform erfüllt ist und kein Nichtschlüsselattribut von einem Schlüsselkandidaten transitiv abhängt.

Anders ausgedrückt:

Eine Relation ist in der dritten Normalform wenn die zweite Normalform vorliegt und kein Nichtschlüsselattribut von einem anderen Nichtschlüsselattribut funktional abhängig ist.

Transitive Abhängigkeit

Beispiel:

In einer Tabelle „Mitarbeiter“ sind uA die Attribute „Name“, „PLZ“ und „ORT“ vorhanden.

- zu jedem Namen gehört eine PLZ
- zu jeder PLZ gehört ein Ort

--> Der Ort ist transitiv (indirekt) abhängig vom Namen.

Mathematisch ausgedrückt:

- Wenn Name --> PLZ und PLZ --> Ort dann Name --> Ort

--> Name --> Ort ist eine transitive Abhängigkeit.

Vor der Normalisierung in 3NF

Zum Verdeutlichen wird der Tabelle CD ein Attribut „GRUENDUNG“ (Gründungsjahr der Band) hinzugefügt.

CD

CD_ID	*INTERPRET*	*GRUENDUNG*	*ALBUM*	*JAHR*
1001	Jeff Beck	1968	Guitar Shop	1989
1002	Monster Magnet	1990	4-Way Diablo	2007
1003	Larry Carlton	1968	Sleepwalk	1982
1004	Monster Magnet	1990	God Says No	2000

Offensichtlich lässt sich der Interpret einer CD aus der CD_ID bestimmen, das Gründungsjahr der Band/Interpreten hängt wiederum vom Interpreten und damit transitiv von der CD_ID ab.

Das Problem ist hierbei wieder die Redundanz. Wird zum Beispiel eine neue CD mit einem existierenden Interpreten eingeführt, so wird das Gründungsjahr redundant gespeichert.

Nach der Normalisierung in 3NF

CD

CD_ID	*ART_ID*	*ALBUM*	*JAHR*
1001	101	Guitar Shop	1989
1002	102	4-Way Diablo	2007
1003	103	Sleepwalk	1982
1004	102	God Says No	2000

ARTIST

ART_ID	*INTERPRET*	*GRUENDUNG*
101	Jeff Beck	1968
102	Monster Magnet	1990
103	Larry Carlton	1968

SONG

CD_ID	*TRACK*	*SONG*
1001	1	Guitar Shop
1001	2	Savoy
1001	3	Behind the Veil
1002	1	Wall of Fire
1002	2	You‘re Alive
1003	1	Last Nite

Die Tabelle wird aufgeteilt, wobei die beiden voneinander abhängigen Attribute in eine eigene Tabelle ausgelagert werden. Der Schlüssel der neuen Tabelle muss als Fremdschlüssel in der alten Tabelle erhalten bleiben.

An der Tabelle „SONG“ wurden keine Änderungen bei der Übertragung in die 3. Normalform vorgenommen. Sie ist nur der Vollständigkeit halber aufgeführt.

Übung

Überprüfen Sie folgende Tabelle. Falls notwendig wenden Sie die 1NF, 2NF und 3NF an.

PERSONAL

PID	*Name*	*Vorname*	*AID*	*Abteilung*	*PrID*	*Projekt*	*Zeit (h)*
1	Doe	John	1	Software	12, 13	Web, Server	5,12
2	Keys	Bill	2	Hardware	13	Server	50
3	Meyers	Liv	1	Software	12, 34	Web, Router	16, 21

Vor der Normalisierung in 1NF

PERSONAL

PID	*Name*	*Vorname*	*AID*	*Abteilung*	*PrID*	*Projekt*	*Zeit (h)*
1	Doe	John	1	Software	12, 13	Web, Server	5,12
2	Keys	Bill	2	Hardware	13	Server	50
3	Meyers	Liv	1	Software	12, 34	Web, Router	16, 21

Kriterien für 1NF:

- Alle Attribute sind atomar √
- Attribute haben keine Wertelisten X
  - „12,13“
  - Web, Server
  - 5,12
  - etc.

Nach der Normalisierung in 1NF

PERSONAL

PID	*NAME*	*VORNAME*	*AID*	*ABTEILUNG*	*PRID*	*PROJEKT*	*ZEIT*
1	Doe	John	1	Software	12	Web	5
1	Doe	John	1	Software	13	Server	12
2	Keys	Bill	2	Hardware	13	Server	50
3	Meyers	Liv	1	Software	12	Web	16
3	Meyers	Liv	1	Software	34	Router	21

Vor der Normalisierung in 2NF

PERSONAL

PID	*NAME*	*VORNAME*	*AID*	*ABTEILUNG*	*PRID*	*PROJEKT*	*ZEIT*
1	Doe	John	1	Software	12	Web	5
1	Doe	John	1	Software	13	Server	12
2	Keys	Bill	2	Hardware	13	Server	50
3	Meyers	Liv	1	Software	12	Web	16
3	Meyers	Liv	1	Software	34	Router	21

Kriterien für 2NF:

- 1NF ist erfüllt √
- Jede Relation bildet nur einen Sachverhalt ab X
  - Personal
  - Abteilung
  - Projekt
  - Zeit

Nach der Normalisierung in 2NF

PERSONAL

PID	AID	*NAME*	*VORNAME*
1	1	Doe	John
2	2	Keys	Bill
3	1	Meyers	Liv

ABTEILUNG

AID	*ABTEILUNG*
1	Software
2	Hardware

PROJEKT

PRID	*PROJEKT*
12	Web
13	Server
34	Router

PERSONAL_PROJEKT

PID	PRID	*ZEIT*
1	12	5
1	13	12
2	13	50
3	12	16
3	34	21

Vor der Normalisierung in 3NF

PERSONAL

PID	AID	*NAME*	*VORNAME*
1	1	Doe	John
2	2	Keys	Bill
3	1	Meyers	Liv

ABTEILUNG

AID	*ABTEILUNG*
1	Software
2	Hardware

PROJEKT

PRID	*PROJEKT*
12	Web
13	Server
34	Router

PERSONAL_PROJEKT

PID	PRID	*ZEIT*
1	12	5
1	13	12
2	13	50
3	12	16
3	34	21

Kriterien für 3NF:

- 2NF ist erfüllt √
- Kein Nichtschlüsselattribut ist von einem Schlüsselkandidaten transitiv abhängig √