Hoofdstuk 3. Normalisatie

3.1 Probleemstelling

Uit de analyses tot nu toe, blijkt welke gegevens moeten worden vastgelegd en wat hun samenhang is. Als nu zonder nader onderzoek deze gegevens worden opgeslagen in bestanden (XML, plain text, databank,. . .), dan kunnen zich tijdens het onderhoud van die bestanden vele problemen voordoen. Het is een doel van data analysis om een zodanige structuur te vinden dat deze problemen niet kunnen optreden. Deze problemen kunnen het best met een voorbeeld worden toegelicht.
Stel dat bijvoorbeeld uit de informatie-analyse van een Project Management Systeem (PMS) blijkt dat er behoefte is aan de informatie in Tabel 1.

Uit Tabel 1. kan worden afgeleid dat de informatiebehoefte betrekking heeft op de volgende gegevens:

- Projectnummer
- Projectomschrijving
- Budget
- Medewerkersnummer
- Naam
- Afdeling
- Uren

Waarbij Medewerkersnummer, Naam, Afdeling en Uren een herhaald aantal keren voor komen.

Als deze gegevens zo opgeslagen worden, dan worden de gegevens van de medewerker herhaald bij ieder project waaraan door die medewerker gewerkt wordt. “005 Fred” in dit voorbeeld.

PROJECTOVERZICHT.

PROJECT	BUDGET	MEDEWERKER	AFDELING	UREN
001 VDU	1000	003 Ton	Analyse	60
		005 Fred	Programmering	100
002 CRT	800	004 Peter	Analyse	200
		005 Fred	Programmering	50
003				

Tabel 1: Een voorstelling van een PMS

Problemen die optreden bij het wijzigen, toevoegen en verwijderen van gegevens.

- De schrijfwijze van de naam (Fred) zou bij elk project anders kunnen zijn, wat niet de bedoeling is.

- Wanneer Fred ontslag neemt moet zijn naam overal verwijderd worden. Het ligt voor de hand dat hierbij vergissingen gebeuren en zijn naam dus op een/meerdere plaatsen blijft staan, wat niet zou mogen.

- Telkens als vb. Fred aan een nieuw project gaat meewerken moet ook weer zijn afdeling (programmering) ingevuld worden. Dit leidt al vlug tot fouten.

- Als een naam verandert (van bv. een afdeling) dan moet die op alle plaatsen aangepast worden, met mogelijke vergissingen tot gevolg.

Een databank waar gegevens nodeloos in worden herhaald, is moeilijk te onderhouden en zal dus al vlug vol fouten staan.

Het is de verdienste van Ted CODD (A Relational model of data for large shared data banks – E.F. Codd – Communications of the ACM – vol.13, pp.377/387 (1970)) dat hij al deze problemen heeft onderkend en een oplossingswijze hiervoor heeft aangereikt.
Deze oplossingswijze staat in de literatuur bekend onder de naam van **"normalisatiestappen van CODD**".
CODD heeft deze oplossingswijze voorgeschreven in het kader van zijn theorie over "relational data bases" en nooit bedoeld voor het bepalen van de gegevensstructuur van een verzameling. Toch is het praktische nut van zijn normalisatiestappen juist hierbij erg groot gebleken.

Normalisatie is een techniek

- waarbij een verzameling zodanig in groepen verdeeld wordt,
- dat er zich geen onregelmatigheden bij het onderhoud van de verzameling meer kunnen voordoen.

Deze techniek van normaliseren wordt best altijd bij gegevensanalyse gebruikt.

3.2 Inleiding - Begrippen.

Met behulp van de normalisatiestappen van CODD, kortweg normaliseren genoemd, zijn we in staat om een willekeurige informatiebehoefte te verdelen in een aantal groepen.

Zo een *genormaliseerde groep* bevat altijd een vast aantal gegevens en kan dus als een tabel worden voorgesteld.
In Tabel 2 zie je de genormaliseerde groep Medewerkers, waarbij het gegeven `Nummer' iedere rij een uniek id geeft of ook wel sleutel genoemd. Alle attributen: Naam, Voornaam en Postcode zijn functioneel afhankelijk van het Nummer (dwz: als we spreken over een ander

(medewerkers)nummer, dan hebben we het over een andere medewerker, en verandert dus de naam, voornaam en postcode).Er zijn geen functionele afhankelijkheden aanwezig tussen de attributen onderling. De groep is dus genormaliseerd' en kan ook als volgt worden weergegeven:

Medewerker (Nummer, Naam, Voornaam, Postcode)

Medewerkers			
Nummer	Naam	Voornaam	Postcode
001	Henk	Jan	3500
002	Peter	Els	3600
003	Ton	Piet	3770
004	Peter	Peter	3740
005	Fred	Ilse	3800

Tabel 2: Voorbeeld van een genormaliseerde groep

Hierbij worden per groep tussen de haken de gegevens opgesomd waaruit de groep bestaat en wordt de sleutel onderstreept. Bij een samengestelde sleutel zullen dus meerdere gegevens onderstreept zijn. Voorbeeld van een groep met samengestelde sleutel:

Geldafhaling (Datum, Tijd, Bankkaart, Bedrag)

Het is de gewoonte om de sleutel het eerst te vermelden, maar noodzakelijk is dat niet.
In tegenstelling tot de voorstellingswijze in tabelvorm kunnen op deze manier niet de afzonderlijke voorkomens elk apart worden aangegeven. Deze laatste voorstellingswijze geeft dus alleen het type aan, terwijl met een tabel de afzonderlijke voorkomens (instanties) getoond kunnen worden.
Meestal is een voorstelling van de verschillende typen voldoende om aan te geven hoe een gegevensverzameling is opgebouwd. Maar soms is een voorstelling van de afzonderlijke voorkomens nodig om dit precies te begrijpen.

Voor we dieper ingaan op normaliseren is het aangewezen uitgebreider in te gaan op sleutels aangezien deze een belangrijke rol spelen bij normalisatie.

3.2.1 Sleutels

3.2.1.1 kandidaat-sleutel

Referentienummers worden vaak gebruikt om dingen te identificeren: factuurnummers, onderdeel codes, serienummers zijn daar algemeen bekende voorbeelden van. Vaak worden combinaties van letters en cijfers gebruikt, bijvoorbeeld bij autonummers. Bij de gegevensverwerking worden sleutels om dezelfde reden gebruikt. Sleutels dienen om records (=rij van data in een database tabel bestaande uit een enkele waarde uit elke kolom) te identificeren zodat er naar gerefereerd kan worden of zodat ze benaderd kunnen worden. Sleutel en sleutelwaarde zijn fundamentele begrippen uit de gegevensverwerking. Het zal daarom duidelijk zijn dat binnen de relationele benadering, faciliteiten moeten bestaan om met sleutels te kunnen werken. In feite hebben relationele systemen speciale regels ten aanzien van sleutels.

Een sleutel moet gevormd worden uit één of meer attributen binnen het record of de relatie.
Een kandidaat-sleutel moet steeds voldoen aan 2 voorwaarden:

- Een kandidaat-sleutel is een combinatie van attributen uit de tabel, waarvoor de attribuutwaarden op een unieke wijze een record van alle andere records uit de tabel onderscheidt. Met andere woorden: een kandidaat-sleutel moet uniek zijn. Het is altijd mogelijk om in een relationeel systeem zo een unieke sleutel aan te wijzen, omdat er geen duplicaat records bestaan.

- Een volgende regel zegt, dat wanneer een willekeurig attribuut wordt weggelaten uit de kandidaat-sleutel, de eigenschap van het uniek identificeren dan verloren moet gaan. Dit betekent dat elke kandidaat-sleutel voldoende attributen moet bevatten om elk record uniek te identificeren, maar dat het geen redundante attributen mag bevatten.

Voor elke tabel in de database moeten we minstens één kandidaat-sleutel definiëren. Vaak is het mogelijk dat we meerdere kandidaat-sleutels kunnen beschouwen, dat wil zeggen: meerdere verschillende combinaties van attributen die een unieke sleutel opleveren. Dit zien we in meer detail in Sectie 1.3.3.
In vele gevallen zullen per tabel meerdere kandidaat-sleutels bestaan. De kandidaat-sleutel die hieruit wordt gekozen ter unieke identificatie van het record noemt men de primaire sleutel.

3.2.1.2 Primaire sleutel (Primary Key)

Een primaire sleutel is een kandidaat-sleutel waarin geen van de elementen een `null-waarde' kan bevatten. Primaire sleutels zijn bedoeld om een eenvoudige en duidelijke representant te zijn van de echte objecten. Pogingen om de waarde van de primaire sleutels te wijzigen, moeten zorgvuldig bewaakt worden. Dit, omdat primaire sleutels weer kunnen voorkomen in andere delen van de database (onderwerp van Sectie 5.2.1.3). De primaire sleutel doet in zo een geval dienst als een verwijzing. Elke verandering aan de primaire sleutel zal in de meeste gevallen betekenen, dat elders voorkomende waarden van de sleutel gewijzigd moet worden.

3.2.1.3 Externe sleutel (Foreign Key)

Een externe sleutel is een veld/attribuut in een relationele tabel die met een kandidaat-sleutel van een andere tabel overeenkomt. De foreign key kan gebruikt worden om te refereren naar records in andere tabellen. Bijvoorbeeld, stel dat we twee tabellen hebben, een tabel Klant die alle klantengegevens bevat en een Order tabel met alle klantenorders. Het is de bedoeling dat alle bestellingen moeten worden gekoppeld aan een klant die al in de tabel Klant voorkomt. Om dit te doen, zullen we een externe sleutel in de Order tabel opnemen en deze laten wijzen naar de primaire sleutel van de tabel Klant.
Een tabel kan meerdere foreign keys hebben en elke foreign key kan naar een andere tabel verwijzen.
Neem terug bovenstaand voorbeeld van Klant en Order. We zouden in Order ook nog kunnen bijhouden welke verkoper deze Order opgenomen heeft. We zullen dan een externe sleutel in de Order tabel laten wijzen naar de primaire sleutel van de tabel Personeel.

Een externe sleutel kan ook verwijzen naar de primaire sleutel in de eigen tabel. Bijvoorbeeld, we kunnen in de tabel Personeel de kolom ‘chef’ hebben. ‘Chef’ verwijst dan naar de primaire sleutel van de tabel Personeel of is null.
Onjuiste foreign / primairy key relaties of het niet handhaven van deze relaties zijn vaak de bron van veel database en “data modellering” problemen.

Opdracht 1: Veronderstel dat de producten in een onderneming voorzien zijn van een kleurcode voor de afmeting en in twee verschillende lengtes worden gemaakt. Bepaal de kandidaatsleutels en de primaire sleutel in volgende tabel.

PROD#	AFMETING	KLEUR	LENGTE
P01	8	ROOD	50
P02	10	GROEN	50
P03	12	BLAUW	50
P04	14	GEEL	50
P05	8	ROOD	100
P06	10	GROEN	100
P07	12	BLAUW	100
P08	14	GEEL	100

Opdracht 2: Duid in onderstaande tabellen de primaire en vreemde sleutels aan.

Schilderij

S_ID	Naam	Artiest	Periode	Waarde	Eigenaar
S01	Vissershuis	A04	1882	16.000.000	Boijmans
S02	De balletles	A02	1872	8.500.000	Louvre
S03	Mona Lisa	A01	1499	75.000.000	Louvre
S04	Namiddag te Oostende	A03	1881	200.000	KMSK

Artiest

A_ID	Naam	Voornaam	Geboren	Gestorven
A01	Da Vinci	Leonardo	1452	1519
A02	Degas	Edgar	1834	1917
A03	Ensor	James	1860	1949
A04	Monet	Claude	1840	1926

Eigenaar

Naam	Plaats	Land
Boijmans	Rotterdam	Nederland
Louvre	Parijs	Frankrijk
KMSK	Antwerpen	België

3.3 Normalisatiestappen

Het normaliseren vindt plaats in drie stappen. Vanuit een ongenormaliseerde informatiebehoefte worden de genormaliseerde groepen bepaald. De stappen vormen een recept, dat -indien goed opgevolgd - altijd hetzelfde eindproduct geeft. Het product is niet afhankelijk van de kok die het bereidt.

De drie stappen van het recept luiden:

1. Verwijder de zich herhalende deelverzamelingen
2. Verwijder de attributen die functioneel afhankelijk zijn van slechts een gedeelte van de sleutel.
3. Verwijder de attributen die ook functioneel afhankelijk zijn van andere (niet-sleutel) attributen.

PROJECTOVERZICHT.

Project	Budget	Medewerker	Afdeling	Chef	Uren
001 VDU	1000	003 Ton	Analyse	Johan	60
		005 Fred	Programmering	Ron	100
002 CRT	800	004 Peter	Analyse	Johan	200
		005 Fred	Programmering	Ron	50
003 TTV	...	...			

Tabel 3: Een ongenormaliseerd model

In al deze stappen wordt gesproken over 'verwijderen'. Dit zou er op kunnen duiden dat gegevens echt verwijderd moeten worden en dus geen deel meer uitmaken van het geheel. Dat is natuurlijk niet de bedoeling, want dan zou met behulp van de genormaliseerde groep(en) nooit meer de oorspronkelijke informatiebehoefte vervaardigd kunnen worden.
Het 'verwijderen' in deze normalisatiestappen houdt in:

- het verwijderen ***uit de oorspronkelijke groep***,
- maar het tegelijkertijd ***creëren van een nieuwe groep***. Er mag dus niets écht verwijderd worden.

Iedere stap heeft slechts betrekking op één groep. Als er meerdere groepen zijn dan moet iedere stap voor alle groepen afzonderlijk worden uitgevoerd.

De drie stappen zullen nu eerst zonder veel toelichting getoond worden. Hiervoor is het inmiddels bekende projectoverzicht iets aangepast. Na deze korte behandeling wordt iedere stap afzonderlijk uitvoerig toegelicht in de hierna volgende paragrafen.

Deze ongenormaliseerde informatiebehoefte in Tabel 3 kan als volgt worden voorgesteld:

Projectoverzicht:	Projectnummer Projectomschrijving Budget Medewerkersnummer Naam Afdeling Chef Uren

Stap 1: Verwijder de zich herhalende deelverzamelingen

De groepen die na deze eerste stap ontstaan worden aangeduid met ***eerste normaalvorm (1NV)***. Deze eerste normaalvorm van het projectoverzicht bestaat uit de volgende eerste normaalvormgroepen:

Project:	<u>Projectnummer</u> Projectomschrijving Budget
Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Naam Afdeling Chef Uren

De oorspronkelijke ongenormaliseerde groep is nu opgedeeld in twee groepen waarin géén deelverzamelingen meer voorkomen die zich herhalen. Per groep is de <u>sleutel</u> onderstreept.

Stap 2: Verwijder de attributen die functioneel afhankelijk zijn van slechts een gedeelte van de sleutel.

Na deze stap wordt gesproken over de ***tweede normaalvorm (2NV)***.
Deze tweede normaalvorm bestaat uit de volgende groepen:

Project:	<u>Projectnummer</u> Projectomschrijving Budget
Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Uren
Medewerker:	<u>Medewerkersnummer</u> Naam Afdeling Chef

Alle attributen zijn in elk van deze groepen functioneel afhankelijk van de volledige sleutel.

Stap 3: Verwijder de attributen die ook functioneel afhankelijk zijn van andere (niet-sleutel) attributen.

Na deze stap wordt natuurlijk gesproken over de ***derde normaalvorm (3NV)***. Deze derde normaalvorm bestaat voor het projectoverzicht uit de volgende groepen:

Project:	<u>Projectnummer</u> Projectomschrijving Budget
Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Uren
Medewerker:	<u>Medewerkersnummer</u> Naam Afdeling
Afdeling:	<u>Afdeling</u> Chef

Niet alle groepen worden in iedere stap nader opgedeeld. Zo is de groep "Project" na de eerste stap niet meer gewijzigd. Voor deze groep geldt dus dat de eerste normaalvorm gelijk was aan de tweede en de derde normaalvorm.

Uit deze waarneming vallen twee zaken af te leiden.

1. dat het normalisatieproces ergens kan stoppen; er is sprake van een zogenaamde ***laatste normaalvorm (LNV)***, daar waar het normalisatieproces stopt voor een bepaalde groep.

2. dat een groep in de eerste normaalvorm kan voldoen aan de tweede en de derde normaalvorm; omgekeerd is het zo, dat iedere groep in 3NV altijd voldoet aan 2NV en iedere groep in 2NV altijd voldoet aan 1NV. De drie normaalvormen zijn deelverzamelingen van elkaar.

In de volgende onderdelen wordt in detail ingegaan op de 3 normaalvormen.

3.3.1 Eerste normaalvorm.

Stap 1: Verwijder de zich herhalende deelverzamelingen.

Dit is een stap, die vaak moeilijker gevonden wordt dan de overige stappen. Het is ook een stap, die niet eenduidig is, dat wil zeggen dat het resultaat verschillend kan zijn. Het eindresultaat van de drie normalisatiestappen is wél eenduidig, maar de af te leggen weg kennelijk niet.

Om enige steun te bieden bij het vinden van de eerste normaalvorm, is ook hiervoor een recept ontwikkeld:

1.1. Inventariseer alle elementaire gegevens.
1.2. Verwijder alle procesgegevens.
1.3. Doe het volgende totdat er géén nieuwe groepen meer ontstaan:
(a) Geef de sleutel van de groep aan.
(b) Geef de deelverzameling aan die een herhaald aantal keren voorkomt.
(c) Herhaal de sleutelgegevens van de oorspronkelijke groep samen met de gegevens van de zich herhalende deelverzameling als een nieuwe groep.
(d) Verwijder de zich herhalende deelverzameling uit de oorspronkelijke groep.

Stap 1.1
Alleen elementaire gegevens mogen geïnventariseerd worden. Voor ieder aanwezig samengesteld gegeven moet dus minstens bekend zijn uit welke elementaire gegevens dit is samengesteld. Ieder elementair gegeven behoort een naam te krijgen, waarmee het zich van alle andere gegevens onderscheidt. Of deze opsplitsing zinvol is, hangt ervan af of we ons zinvolle bewerkingen op de afzonderlijke componenten kunnen voorstellen.

Vb1: moet straat en huisnummer als één gegeven worden beschouwd of niet?
Vb2: de score van een voetbalwedstrijd kan worden beschouwd als een samengesteld gegeven, namelijk een geordend paar van twee natuurlijke getallen, maar als het berekenen van het totaal aantal gescoorde doelpunten in één seizoen belangrijk is maak je hier best twee elementaire gegevens van.

Stap 1.2
Soms kan een gegeven door berekening uit andere gegevens worden afgeleid. In feite is er dan geen sprake van een keuze uit een verzameling: de keuze ligt vast na het maken van keuzes voor andere gegevens. Dergelijke gegevens noemen we berekende gegevens of procesgegevens. Ze kunnen integraal deel uitmaken van de informatiebehoefte van een bedrijfsproces. Procesgegevens moeten wel apart genoteerd worden, maar worden in de normalisatiestappen niet meegenomen mits aan de volgende voorwaarden is voldaan:

- Alle voor de berekening benodigde gegevens zijn aanwezig.
- De voor de berekening benodigde gegevens bevatten op het tijdstip waarop de berekening moet worden uitgevoerd nog de juiste waarden.

In ons voorbeeld in tabel 3 zijn geen procesgegevens aanwezig. Voorbeelden van procesgegevens zijn:

- de dag van de week kan worden berekend uit de datum, maar is niettemin nuttig om te bepalen of het om een verlofdag gaat.
- ‘aantal calorieën’ is afleidbaar uit ‘hoeveelheid vetten’, ‘hoeveelheid koolhydraten’ en ‘hoeveelheid eiwitten’, als je weet dat deze drie energiebronnen respectievelijk 9kcal/g, 4kcal/g en 5kcal/g leveren. Nochtans staan op de verpakking van veel voedingswaren alle vier de gegevens vermeld.
- ‘aantal dienstjaren’ is berekenbaar uit ‘datum indiensttreding’, op voorwaarde dat je de huidige datum als bekend veronderstelt.

Stap 1.3
Zich herhalende deelverzamelingen kunnen genest voorkomen. D.w.z. dat er een zich herhalende deelverzameling bevindt binnen een andere deelverzameling, die ook een herhaald aantal keren voorkomt. De eerste normalisatiestap moet zover worden doorgevoerd totdat alle zich herhalende deelverzamelingen zijn opgedeeld.

Stap 1.3(a)
Deze stap is bepalend voor de wijze waarop de eerste normalisatiestap verloopt. Afhankelijk van de keuze van de sleutel zullen er al of niet (geneste) herhalingen voorkomen. Deze wellicht wat cryptische woorden zullen we aan de hand van het voorbeeld nader toelichten.

Na de inventarisatie van de elementaire gegevens is het projectoverzicht als volgt voorgesteld:

Projectoverzicht:	Projectnummer Projectomschrijving Budget Medewerkersnummer Naam Afdeling Chef Uren

Er is blijkbaar gekozen voor het gegeven Projectnummer als sleutel. Want de eerste normaalvorm wordt gevormd door twee groepen, Project en Medewerker. Men had ook de samengestelde sleutel Projectnummer en Medewerkersnummer kunnen kiezen, maar dan waren er geen zich herhalende deelverzamelingen meer aanwezig geweest.

Stap 1.3(b)
Afhankelijk van de sleutelkeuze in stap 1.3(a) zullen nu nul, één of meer zich herhalende deelverzamelingen aanwezig zijn.

Projectoverzicht:	<u>Projectnummer</u> Projectomschrijving Budget Medewerkersnummer Naam Afdeling Chef Uren	komt een herhaald aantal keer voor

Stap 1.3(c)
Nu wordt een nieuwe groep gevormd. Deze moet bestaan uit:

de zich herhalende deelverzameling
+
de sleutel van de oorspronkelijke groep.

Deze laatste wordt in de nieuwe groep opgenomen om de koppeling met de oorspronkelijke groep in stand te houden. Zo krijgen we de volgende twee groepen:

Projectoverzicht:	<u>Projectnummer</u>	
	Projectomschrijving	
	Budget	
	Medewerkersnummer	herhaald
	Naam	
	Afdeling	
	Chef	
	Uren	
Projectmedewerker:	***Projectnummer***	
	Medewerkersnummer	
	Naam	
	Afdeling	
	Chef	
	Uren	

Let op:

- Op dit ogenblik is de oorspronkelijke groep nog compleet.
- In de nieuwe groep is nog géén sleutel bepaald.

Stap 1.3(d)
Nu wordt pas de oorspronkelijke groep aangepast en vindt de eigenlijke '*verwijdering*' plaats. Door dit in deze stappen te doen kan men zich ervan overtuigen dat alle gegevens die verwijderd worden ook daadwerkelijk in de nieuwe groep zijn opgenomen. De sleutel, die in de nieuwe groep herhaald is, wordt natuurlijk niet verwijderd.

Project:	<u>Projectnummer</u>
	Projectomschrijving
	Budget
Projectmedewerker:	***Projectnummer***
	Medewerkersnummer
	Naam
	Afdeling
	Chef
	Uren

Stap 1.3(a) (2de iteratie)
De keuze van de samengestelde sleutel 'Projectnummer' en 'Medewerkersnummer' is misschien niet direct duidelijk. Indien alleen Medewerkersnummer als sleutel gekozen was, dan kwamen Uren en Projectnummer nog een herhaald aantal keren voor. Bij de nu gekozen sleutel zijn er géén zich herhalende deelverzamelingen meer aanwezig. Stap 1 van het normalisatieproces is voltooid en de eerste normaalvorm is bepaald.

Project:	<u>Projectnummer</u> Projectomschrijving Budget
Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Naam Afdeling Chef Uren

3.3.2 Tweede normaalvorm.

Stap 2: Verwijder de attributen die functioneel afhankelijk zijn van slechts een gedeelte van de sleutel.

Om te komen van de eerste tot de tweede normaalvorm, moeten de attributen die slechts van een *gedeelte van de sleutel* afhankelijk zijn in een aparte groep worden opgenomen. Alleen groepen met een samengestelde sleutel komen hiervoor in aanmerking, want alleen bij een samengestelde sleutel kan een attribuut afhankelijk zijn van een gedeelte van de sleutel. Eigenlijk is de sleutel van groepen, die nog niet in tweede normaalvorm zijn, geen goede sleutel voor alle attributen, want een sleutel mag volgens de definitie géén overtollige gegevens bevatten. Voor sommige attributen, is dat echter wel het geval. Het zijn dan ook deze attributen die samen met het deel van de sleutel dat voor hun geen overtollige gegevens bevat, een afzonderlijke groep gaan vormen.

Het recept voor de tweede normaalvorm luidt als volgt:

2.1. Geef de attributen aan die niet functioneel afhankelijk zijn van de volledige sleutel.
2.2. Vorm een aparte groep voor ieder deel van de sleutel waarvan attributen functioneel afhankelijk zijn.
2.3. Neem in iedere groep de attributen met het bijbehorende sleuteldeel op.
2.4. Verwijder deze attributen uit de oorspronkelijke groep.

Stap 2.1
Om dit te kunnen doen moet van ieder attribuut de vraag beantwoord worden:
"Welk gegeven of combinatie van gegevens identificeert dit attribuut op een éénduidige wijze?"
In het voorbeeld "Projectoverzicht" komt de *groep Project* niet in aanmerking, daar die groep niet beschikt over een samengestelde sleutel. Binnen de *groep Projectmedewerker* zijn o.a. de volgende functionele afhankelijkheden te onderkennen:

Attribuut Naam	is functioneel afhankelijk van sleutel	Medewerkersnummer
Attribuut Afdeling	is functioneel afhankelijk van sleutel	Medewerkersnummer
Attribuut Chef	is functioneel afhankelijk van sleutel	Medewerkersnummer
Attribuut Uren	is functioneel afhankelijk van sleutel	Projectnr + Medewerkersnr

De attributen Naam, Afdeling en Chef zijn slechts functioneel afhankelijk van een deel van de sleutel, het Medewerkersnummer.

Stap 2.2
Het kan gebeuren dat een samengestelde sleutel in meerdere delen gesplitst kan worden en dat van ieder deel afzonderlijk attributen functioneel afhankelijk zijn. Er moeten dan meerdere groepen gevormd worden.
In ons voorbeeld ontstaat slechts één nieuwe groep: Medewerker.

Stap 2.3
De in stap 2.2 geïnventariseerde groepen moeten nu gevuld worden. Dit moet zodanig gebeuren, dat iedere nieuwe groep voldoet aan de eisen van de tweede normaalvorm:

- er mogen dus géén herhalingen aanwezig zijn
- en alle attributen moeten functioneel afhankelijk zijn van de volledige sleutel.

In het voorbeeld ontstaat nu de groep Medewerker

- met als sleutel Medewerkersnummer,
- en als attributen Naam, Afdeling, Chef.

Na deze stap is de oorspronkelijke groep Projectmedewerker opgebouwd uit de volgende groepen:

Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Naam Afdeling Chef Uren
Medewerker:	<u>Medewerkersnummer</u> Naam Afdeling Chef

Stap 2.4
Doordat de oorspronkelijke groep nog intact is, kunnen we heel zorgvuldig te werk gaan bij het vervangen van de attributen uit deze groep. De sleutel van de oorspronkelijke groep mag niet aangetast worden.

We hebben nu in totaal de volgende groepen gekregen:

Project:	<u>Projectnummer</u> Projectomschrijving Budget
Projectmedewerker:	<u>Projectnummer</u> <u>Medewerkersnummer</u> Uren
Medewerker:	<u>Medewerkersnummer</u> Naam Afdeling Chef

3.3.3 Derde normaalvorm.

Stap 3: Verwijder attributen die ook functioneel afhankelijk zijn van andere (niet-sleutel) attributen.

Bij de stap naar de derde normaalvorm moeten de attributen

- die functioneel afhankelijk zijn van de volledige sleutel (2NV),
- maar ook nog functioneel afhankelijk zijn van andere attributen,

in aparte groepen worden opgenomen.

Dit soort afhankelijkheden, tussen attributen onderling, wordt wel aangeduid als *transitieve afhankelijkheden*.
Uiteraard kunnen alleen groepen met meerdere attributen deze soort afhankelijkheden bevatten.
De afhankelijkheid tussen de attributen onderling moet wél een functionele afhankelijkheid zijn om over te gaan tot de vorming van een nieuwe groep. Afhankelijkheden, waarbij het ene attribuut op de een of andere manier samenhangt met een ander attribuut, zijn hiervoor geen reden. Een voorbeeld van dit soort (losse, toevallige) afhankelijkheden is het feit dat de datum-in-dienst groter moet zijn dan de geboortedatum van een medewerker. Dit is echter géén functionele afhankelijkheid.

De *afhankelijkheden* die *transitief* genoemd worden zijn altijd van het type 'sleutel attribuut' en deze moeten in een aparte groep worden opgenomen.

Het recept voor de derde normaalvorm is het volgende:

3.1. Geef de attributen aan die ook functioneel afhankelijk zijn van andere (niet-sleutel) attributen.
3.2. Vorm een aparte groep voor ieder attribuut of combinatie van attributen, waar andere attributen functioneel van afhankelijk zijn.
3.3. Neem in iedere nieuwe groep de attributen met hun bijbehorende sleutel op.
3.4. Verwijder de attributen van de nieuwe groep(en) uit de oorspronkelijke groep.

Stap 3.1
Om dit te kunnen doen moet van ieder attribuut worden vastgesteld of er nog één of meer andere attributen zijn die het beschouwde attribuut uniek identificeren.

In het voorbeeld "Projectoverzicht" is in de groep Project geen functionele afhankelijkheid aanwezig tussen de attributen Projectomschrijving en Budget. De groep Projectmedewerker bevat slechts één attribuut en komt dus niet in aanmerking. Maar in de groep Medewerker:

- Is het attribuut Chef functioneel afhankelijk van het attribuut Afdeling.

- de attributen Naam en Afdeling zijn onderling onafhankelijk.

In feite is hier vastgesteld dat iedere medewerker één chef heeft en één afdeling. Maar tevens dat iedere afdeling ook één chef heeft en dat dus de chef van de medewerker de chef is van de afdeling van die medewerker.

Stap 3.2
Er kunnen meerdere transitieve afhankelijkheden aanwezig zijn. Alle unieke attributen of attribuutcombinaties, die als sleutel fungeren in transitieve afhankelijkheden, geven aanleiding tot de vorming van verschillende groepen.

In het voorbeeld ontstaat slechts één nieuwe groep: Afdeling.

Stap 3.3
De in stap 3.2 geïnventariseerde groepen moeten nu van sleutels en attributen worden voorzien. De op deze manier nieuw gevormde groepen moeten wél aan de definitie van de derde normaalvorm voldoen:

- zij mogen dus geen herhalingen bevatten
- en alle attributen moeten functioneel afhankelijk zijn van de volledige sleutel
- en onderling geen functionele afhankelijkheden bevatten.

De nieuw gevormde groep Afdeling bestaat uit twee gegevens, "Afdeling" en "Chef". Wat is nu de sleutel: Afdeling of Chef?

Een sleutel moet éénduidig identificeren en dus unieke waarden bevatten. Er kunnen zich verschillende situaties voordoen. De gebruiker zal moeten aangeven welke situatie voor hem geldig is.

<u>Situatie a:</u>
Iedere afdeling heeft één chef en iedere chef is chef van één afdeling.

Afdeling	Chef
A	1
B	2
C	3

Door het tekenen van een tabel met afzonderlijke voorkomens is direct te zien dat er sprake is van twee kandidaatsleutels. Welke er gekozen wordt is niet belangrijk.

<u>Situatie b:</u>
Iedere afdeling heeft één chef, maar een chef kan van meerdere afdelingen chef zijn.

Afdeling	Chef
A	1
B	2
C	1

Het enige gegeven dat nu een unieke waarde bevat is Afdeling. Dit is dus de sleutel in deze situatie.

Situatie c:
Iedere afdeling heeft meerdere chefs, maar iedere chef is slechts chef van één afdeling.

Afdeling	Chef
A	1
A	2
B	3
B	4

Nu is Chef de sleutel.

Situatie d:
Iedere afdeling heeft meerdere chefs en iedere chef kán chef zijn van meerdere afdelingen.

Afdeling	Chef
A	1
A	2
B	3
B	1

Geen van beide gegevens bevat nu nog unieke waarden. De combinatie van beide gegevens is wél uniek. Beide gegevens vormen een samengestelde sleutel.

De gebruiker heeft aangegeven dat in zijn omgeving situatie a geldt, maar dat situatie b niet uitgesloten wordt geacht. Om deze reden is gegeven Afdeling tot sleutel gekozen.

Na deze stap is de oorspronkelijk 2NV groep Medewerker nu opgebouwd uit de volgende groepen:

Medewerker:	Medewerkersnummer Naam Afdeling Chef
Afdeling:	Afdeling Chef

Stap 3.4
Uit de nu nog intact zijnde oorspronkelijke groep

- moeten de attributen van de nieuw ontstane groep(en) verwijderd worden.
- en de sleutel van de nieuwe groep blijft als attribuut in de oorspronkelijke groep gehandhaafd.

Dit geeft de volgende groepen, die elk op zich in de 3-de normaalvorm zijn:

Project:	Projectnummer Projectomschrijving Budget
Projectmedewerker:	Projectnummer Medewerkersnummer Uren
Medewerker:	Medewerkersnummer Naam Afdeling
Afdeling:	Afdeling Chef

3.3.3.1 Boyce-Codd-normaalvorm (BCNV)

Er zijn ook nog verdere normaalvormen. Voor ons volstaat de 3NV, BCNV is nog beter, maar zeker geen must voor deze cursus. Daarnaast bestaat ook nog 4NV en 5NV, deze zijn niet algemeen geaccepteerd en wordt daarom verder buiten beschouwing gelaten.
Een relatie is in BCNV als deze in 3NV is en er zijn geen transitieve afhankelijkheden, dus geen enkele sleutel bevat informatie over een andere sleutel binnen dezelfde tabel, behalve over de gehele primaire sleutel.

Zie onderstaande tabel uit een database met aannemers:

Aannemers		
Naam	**Werk**	**Aannemer**
Janssen	Loodgieter	B.V. De Loden Gieter
Zeilstra	Loodgieter	B.V. De Loden Gieter
Zeilstra	Elektromonteur	Elektrobedrijf B.V.

In dit voorbeeld wordt aangenomen dat een werknemer niet bij twee aannemers hetzelfde werk mag doen, maar dat hij wel verschillende werkzaamheden mag uitvoeren bij verschillende bedrijven. In deze tabel zijn twee mogelijkheden voor primaire sleutels, namelijk de combinatie van Naam en Aannemer en de combinatie van Naam en Werk. In beide gevallen zal het overgebleven niet-sleutelattribuut informatie bevatten over een deel van de primaire sleutel en niet over de gehele primaire sleutel.

Diezelfde data uit een database met aannemers in BCNV:

Werknemers	
Naam	**Aannemer**
Janssen	B.V. De Loden Gieter
Zeilstra	B.V. De Loden Gieter
Zeilstra	Elektrobedrijf B.V.

Aannemers	
Werk	**Aannemer**
Loodgieter	B.V. De Loden Gieter
Elektromonteur	Elektrobedrijf B.V.