Normaliseren

Een van de voornaamste rollen in een informatie systeem is het bewaren van gegevens en liefst over een 'lange tijd'. Meestal doen we dat door middel van een gegevensbank of databank. Deze gestructureerde, digitaal bewaarde verzameling van gegevens is opgezet om de gebruiker te voorzien van een bepaalde informatie behoefte.

En databank is wenselijk als bepaalde vragen worden ingelost:

-moeten deze gegevens frequent worden opgevraagd?
-de zoekinspanning van de gebruiker naar bepaalde gegevens moet minimaal zijn
-er is een noodzaak tot het antwoorden op complexe vragen, meer dan het opvragen van gegevens.

Als er besloten is een database op te stellen dient men een analyse van de informatiegegevens te volbrengen.

1. Analyse van de informatie gegevens:
   De benodigde aspecten voor dit zijn:
   - Er moetduidelijkheid zijn over de omvang en aard van het **domein** waar de informatie zal opgevraagd worden. De onderwerpen in een domein worden **entiteiten** genoemd
   - Er moet ingeschat worden welke vragen en de frequentie waarop ze zullen voorkomen
   - Het verachte aantal gebruikers van de database.
   - Het verwachte aantal software pakketten dat deze databank zal aanspreken. En eventuele nadelige gevolgen voor de gebruikers
   - Wat zijn de privileges en aan wie worden deze toevertrouwd. Wat is de eventuele private en publieke data
   - Wie mag er data invoeren of veranderen of eventueel wissen
   - Wat is de duurzaamheid van de gegevens
   - Er moet ingeschat worden hoe duurzaam de structuur van de gegevens is. Een DBS (database systeem) is onderworpen aan een snelle verandering van gegevens maar wel in dezelfde structuur van de database. Als de informatiebehoefte veranderd kan het soms nodig zijn de gegevens **structuur** van de database te veranderen. Daarom moet het mogelijk zijn deze structuur aan te passen.
2. Datamodel
   Een model is een abstracte voorstelling van de werkelijke wereld. Daarin bedoelen we dat de maker van een model bepaalde noodzakelijke kenmerken van de werkelijkheid isoleert, en alle andere kenmerken als overbodig catalogeert en achterwege laat.
   In een databank werkt men in hoofdzaak met data of gegevens. Daarom zullen we in ons geval een gegevens- of datamodel moeten maken. Dit zal een abstracte voorstelling zijn van het informatie systeem waarbij alle dynamische gedeelten achterwege blijven. We zullen ons dan ook toespitsen op statische informatiestructuren. We moeten ook het belang onderkennen van hun onderlinge verbanden en eventuele beperkingen.
   Zoals voordien reeds benadrukt gaat het hier over de structuur van de gegevens. Wat hun inhoud is doet niet direct iets ter zake. Een gegevens model uitgewerkt voor een hobbyclub kan specificeren dat ze de namen, adressen en inschrijvingen in de club opslaat. Wat die

namen zijn of van waar ze komen is niet direct van belang. Dat heeft niets te maken met de structuur van het model. Wel kan het van belang zijn dat een adres welke opgenomen wordt moet voor zien zijn van een postnummer. Van welk land is terug niet van belang als het niet gespecificeerd wordt.
Hoe de gegevens zullen gebruikt worden zullen we dan wel vastleggen in een ander model. Dit noemen we dan het procesmodel. Dit model zal op zijn beurt alle dynamische aspecten in beeld brengen.
Het gegevens- of datamodel heeft dus enkel als doel om de structuren van de database aan te geven.
Het doel van dit ie een eenduidige voorstelling te krijgen van de structuren en connecties die aan de grondslag liggen van de verschillende gegevensstromen. Daardoor zal dit model een zeker communicatie middel zijn tussen de belanghebbenden , ontwerpers en beheerders enerzijds en de gebruikers anderzijds.

3. Database systemen
   Als men een **datamodel** gemaakt hebben dienen we dit om te zetten naar een **dataschema** die verbonden is aan een gegeven **datasysteem**. Een datasysteem is de samenstelling van een datamanagement systeem (**DBMS**) en een database systeem (**DBS**). Het data bank systeem dat wij gebruiken is MS- Access. Het structuur type is dan ook **Relationeel.**
4. Normaliseren
   Om de indeling van een databank te bepalen bestaan er verschillende technieken. Eén van die technieken is de **normalisatie.** Deze is uitgebracht door de grondlegger van het relationele model, mr E.Codd (1970), en hetdoel ervan was het vermijden van redundantie. Redundantie van gegevens is het op verschillende plaatsen verschijnen van de zelfde informatie. Aan het feit van redundantie zijn er nadelen nl:
   - Risico van inconsistentie – elkaar tegenspreken van informatie
   - Moeilijker worden van onderhoud van de informatie

   Bij **normalisatie** zullen we in een aantal stappen (4) de gegevens **uiteenrafelen** en dan **indelen** en **opslaan** in een **beperkt** aantal overzichtelijke tabellen, die **relationeel** tot elkaar staan.
5. Eerste stap: **de nulde normaalvorm** (0nv)
   Het normaliseren van gegevens hebben steeds al uitgangspunt de informatie behoefte van de gebruiker van de database. De indeling en de inhoud van de tabellen van de database wordt bepaald door informatie die de gebruiker wenst te zien en te gebruiken.
   Om het ons gemakkelijk te maken zullen we de uitleg koppelen aan een voorbeeld.
   Neem aan, en dat is wel reëel, op een radio zender worden verschillende liedjes gepresenteerd. De artiesten die de liedjes brengen krijgen bij iedere maal dat ze gespeeld worden een kleine vergoeding van SABAM.(vereniging voor auteurs en componisten). Daarom vraagt men dat de platenruiters iedere maand van hun programma's een lijst inleveren van alle nummers welke hij of zij die maand gespeeld hebben. Men heeft daarvoor in het radio station en databank met alle liedjes die men heeft in het gebouw. Daaruit distilleert elke persoon per uitzending een lijst elke op de server geplaatst wordt. Maandelijks maakt men dan een rapport welke verstuurd wordt naar SABAM die dan de berekening maakt.
   De lijst bestaat uit een code, (bepaald in overleg), datum, titel, artiest, soortcode, soort omschrijving.

rd

datum	1/1/2000			totaal	1
	Code	Titel	Artiest	Soortcode	Soortomschrijving
	10	marieke	j.Brel	S	Single

Figuur 1

In bovenstaande figuur ziet u een voorbeeld van een lijst welke moet opgestuurd worden naar SABAM.

Hoe komt men nu aan de nulde normaalvorm. We hebben een bepaald soort gegevens dat we kunnen afleiden uit dit overzicht namelijk in eerste instantie de **vaste of constante gegevens**, zoals bijvoorbeeld de datum. En ten tweede de gegevens die zich herhalen door de lijst door de zogenaamde **herhalende of repeterende** gegevens. Ook als derde hebben we nog de **procesgegevens.** Deze laatste krijgen we door berekeningen of een rekenproces. Dit wordt nooit opgenomen in een database.

We werken dus met volgende gegevens, datum, code, titel, artiest en soort. De aanduiding soort is geen **elementair** element. Daarin bedoelen we twee gegevens namelijk soortcode (S, C, B, L) en soortomschrijving(Single, CD, Band,LongPlay). We moeten dus het gegevens splitsen in 2 elementaire gegevens. Het splitsen krijgen we enkel bij **samengestelde gegevens.**

We houden dus de elementaire gegevens over nl: datum, code, titel, artiest, soortcode en code omschrijving. Deze gegevens noemen nu **attributen**.

Als we deze 'attributen' nu bekijken dan valt er iets in op. Het eerste attribuut, datum, komen we maar 1 maal tegen en de andere gegevens komen herhaald tegen. We noteren dan dit als volgt:

datum, RG(code, titel, artiest, soortcode, soortomschrijving)

daarbij staat RG voor repeterende groep (Repeating Group). Dit zijn gegevens die meerdere keren voorkomen, en staan tussen ronde hakjes geplaatst. Uit figuur1 kunnen we afleiden dat de datum 1 maal voorkomt en dan de andere velden per song herhaalt worden. Als we een ganse lijst van een maand bekijken valt het op dat de datum een reeks van gegevens vooraf gaat en dat de gegevens gekoppeld zijn aan die datum. M.a.w de datum geeft ons toegang tot precies 1 overzicht . Daarom kan men stellen dat dit een sleutel gegeven of sleutelattribuut. We geven dit aan door het te onderstrepen. We hebben dus de nulde normaalvorm gevonden

0NV → (<u>datum</u>,RG(code, titel, artiest, soortcode, soortomschrijving))

Opmerkingen:

- Sleutel attribuut moet altijd een **waarde** hebben
- Sleutel attribuut is **steeds uniek**

rd

- Sleutel attribuut moet minimaal zijn
- Soms kunnen meerdere attributen als sleutel attribuut fungeren. Dit wordt dan kandidaat-sleutels genoemd. In het algemeen kiest men uit de kandidaat-sleutels 1 die zal fungeren als sleutel attribuut.

6. Tweede stap: **eerste normaal vorm(1NV)**

In de vorige 0 NV zitten een paar onvolmaaktheden. Neem nu dat een nummer in deze lijst meer dan 1 keer gedraaid is. In dit geval zal steeds de gegevens moeten herhaalt worden. Neem aan dat het liedje 20 x gedraaid wordt moet dat 20 herhaalt worden. De vraag rijst kan dat niet 1 x genoteerd worden met een aantal aanduiding? Natuurlijk moeten we dan een ander veld het aantal keren dat het gedraaid wordt aanduiden. In de eerste normaal groep gaan proberen dit euvel te verhelpen.

We bepalen 1NV door de repeterende groep apart te nemen en uit te breiden met de sleutel van de 0NV. Hierna bepalen we voor de nieuwe groep een sleutel attribuut.

De RG bestaat uit (code, titel, artiest, soortcode, soortomschrijving). Deze nemen we apart en voegen de sleutel datum eraan toe. We krijgen dus:
(datum, code, titel, artiest, soortcode,soortomschrijving)

Let op: alhoewel datum een sleutel attribuut was in de 0NV en daardoor onderstreept kunnen we deze niet terug zomaar onderstrepen. We gaan dus een nieuwe sleutel aanwijzen. Om dit te doen moeten we ons eerst afvragen hoe we de nieuwe groep zien in het originele overzicht. Zo kunnen we bepalen wat de goede sleutelkeuze is.

Onze groep gegevens, met eenmaal een datum, eenmaal een code, eenmaal een titel,… vinden we terug als 1 regel van het overzicht. Om precies één regel aan te duiden moeten we eerst aanduiden welk formulier, van welke datum we willen bekijken. We selecteren één regel door het aanduiden van de code. Bij het nemen van die code, we nemen aan dat dezelfde persoon per uitzending maar één maal dezelfde artiest speelt. Dus als men per datum een code aanduid ziet men hoeveel maal dat liedje per maand gespeeld is. In dit geval zal er en combinatie van attributen de sleutel aanduiden dus:

(<u>datum</u>, <u>code</u>, titel, artiest, soortcode, soortomschrijving)

Omdat de sleutel nu bestaat uit twee delen en daar de sleutel altijd minimaal moet zijn, moeten we even controleren als de tweede sleutel voldoet aan de gestelde normen. Dus we gaan na als alleen de datum als sleutel voldoet. Dat kan niet. Bij één datum kan het zijn dat 1 persoon bij één datum meerdere nummers gedraaid heeft. Dus we hebben meerdere regels en we zoeken precies één regel.

Kan dat met code? Dat zou zo zijn maar enkel als de persoon het liedje maar 1 x draait die dag. We kunnen aannemen dat dit niet zo is dus het kan zijn dat de persoon per maand het liedje enkele malen draait, dus we kunnen dezelfde code terug vinden op verschillende datums, dus de code alleen voldoet ook niet.

rd

Om de 1NV te bepalen starten we van de 0NV en verwijderen we de repeterende groep en kijken wat overblijft. Dit voegen we toe aan de als aparte groep toe.

0NV: (datum,RG(code,titel,artiest,soortcode,soortomschrijving))

Verwijderen we de RG dan houden we enkel (datum) over.

Deze groep voegen we toe en krijgen de eerste normaal vorm 1NV

(datum)

(datum, code, titel, artiest,soortcode, soortomschrijving)

We zien hier dat we geen RG niet meer over hebben dus we hebben 1 deel van de vraag opgelost. Het tweede deel is als de persoon dit vaker draait, hij moet het nog steeds opnieuw invoeren

Let op: we zien in het voorbeeld dat de dat de datum ook een onderdeel is van het sleutel attribuut van de 1 NV. Dit is vaak zo maar niet altijd. Let dus goed op voor de sleutel!

7. Derde stap: **tweede normaalvorm (2NV)**

Om tot deze normaal vorm te komen onderzoeken we in 1NVals er attributen zijn die **niet** tot de sleutel behoren (niet onderstreept) en die niet van de gehele sleutel afhankelijk zijn, maar slechts van een gedeelte van de sleutel. We kijken even naar ons voorbeeld. Daaw we om deze normaal vorm te bepalen we enkel de 1NV nodig hebben kijken we uitsluiten daar naar:

(datum, code titel, artiest, soortcode, soortomschrijving)

We zoeken nu naar de niet sleutel attributen die slechts afhankelijk zijn van een deel van de sleutel. Dus bij ons allen datum of alleen code. De gegevens soortcode, soortomschrijving hebben in feite niets te maken met de datum, maar zijn wel afhankelijk van de code van het gedraaide nummer, want geef me de code en je kan die attributen bepalen. Dus deze gegevens of attributen zijn afhankelijk van een deel van de sleutel

(code, titel,artiest, soortcode, soortomschrijving)

In de orginele groep laten we de betreffende niet sleutelvelden achterwege en krijgen dan:

(datum, code)

let er echter op dat het sleuteldeel waarvan de niet sleutelvelden afhankelijk zijn gewoon blijven bestaan. Dus we krijgen dan:

(code, titel, artiest, soortcode, soortomschrijving)

(datum, code)

We vullen dit aan met de groepen die we reeds voordien gevonden hadden in 1NV maar waar geen samengestelde sleutel aanwezig was. En krijgen dan als tweede normaalvom:

rd

(code, titel, artiest, soortcode, soortomschrijving)

(datum , code)

(datum)

We hebben nu veel redundantie (overtolligheid) verwijderd. Doordat in de eerst groep de code de sleutel is, ieder nummer wordt daar hooguit 1 maal in op genomen. Wordt dat nummer echter vaker gespeeld zal het vaker opgenomen worden in de tweede groep(datum, code). Doordat de gegevens reeds gekend zijn in de eerste groep worden ze niet nog eens opgenomen.

8. Stap 4: **derde normaalvorm (3NV)**

In de laatste normaal groep zit er nog steeds overtollige gegevens, die te maken hebben met de attributten soortcode en soortomschrijving. We weten bijvoorbeeld voor soortcode we S, C, B,L hebben. Deze staan voor single, CD, Band ? LongPlay. Neem nu even aan dat we 1000 nummers hebben op single. Steeds moeten we de soortcode (S) vernomen. Daardoor hebben we terug een redundantie. Door deze eruit te halen hebben we de 3NV.

Om deze te bepalen vragen we ons af of er niet sleutelattributen zijn die niet afhangen van de sleutel maar van andere niet sleutel attributen. Indien dat het geval is sla deze op in een nieuwe groep.

Laat ons eens terug gan naar ons voorbeeld:

Om de titel van een nummer te weten hebben we niets aan de artiest. Immers dze zal welmeerdere nummers op zijn repertoire staan hebben, de drager (Singel , cd…) heeft daar ook geen belag bij. De titel is wel afhankelijk van de code of van de sleutel. Dit laten we dan rusten. Hetzelfde met artiest en soortcode en dit om dezelfde reden. Met de soortomschrijving echter is er iets meer aan de hand. Om deze te weten hoeven we enkel te weten welke soortcode van toepassing is. Immers geef me de soortcode en ik vertel wat de soortomschrijving is. Deze heeft dus niets te maken met de sleutel maar wel met een ander **niet sleutel attribuut,** en daarom nemen we die op in een aparte groep.

(soortcode, soortomschrijving) waarbij soortcode de sleutel zal worden

(soortcode, soortomschrijving)

In de originele groep laten we afhankelijke niet sleutel attribuut achtewege en houden dan over

(code, titel, artiest, soortcode)

Let wel ondanks dat het attribuut sleutel geworden is in de nieuwe groep, in de oorspronkelijke groep het blijft bestaan.

Door het originele aan te vullen bekomen we de derde normaal vorm

rd

3 NV =

(soortcode, soortomschrijving)

(code, titel, artiest, soortcode)

(datum, code)

(datum)

Het is nu duidelijk dat de derde normaalvorm het uiteindelijke doel geweest is van deze normalisatie. Daar we dit gevonden hebben kunnen we dat nu op eenvoudige wijze opnemen in de database. Iedere groep kan vertaald worden in een afzonderlijke tabel en hierdoor kan men op relatief eenvoudige wijze de gegevens invoeren.

Gebruikte werken:
Informatie analyse volgens NIAM
Informatie analyse volgens SDM.
Systeem ontwikkeling volgen JSD