FDFD: Formal Derivations of Functional Dependencies

Maarten M. Fokkinga

Version 0.951 of 27 oktober 2004, 14:59

Abstract. We show that, and how, functional dependencies can be *formally derived* from a case description in natural language.

Introduction In all of the examinations of the course Databases (Gegevensbanken, 211074) there is an exercise that tests whether the students have the ability to discover functional dependencies in a case that is formulated in natural language. In order to minimize the time to mark the solutions, I've recently asked the students to *decide the truth for each of a given list* of possible functional dependencies, and to motivate their decision. During marking I found out that a formal approach to this problem is quite well possible: derive functional dependencies in a formal way from the "axioms" that form the immediate transliteration of the individual statements in a case description given in natural language. I will explain this formal approach using, and elaborating, a recent exam question.

Figure 1 on page 4 presents the exercise; Figure 2 on page 5 gives the solution that I had in mind. Have a glance at them before reading further.

The formal approach The idea is that we start *as soon as possible* with a formalization of the natural language sentences, and then proceed further with formal calculations only. In this way, the chance for errors due to informalities is minimized. The chance for errors in the formal calculation is minimized too if we invoke machine support for these calculations — which is quite well possible! Thus we propose to write the given sentences in predicate logic notation, transform them into functional dependencies, and finally prove or disprove the proposed functional dependencies by formal calculation. Actually, for the exam question at hand we shall skip the predicate logic notation and immediately formulate the given statements as functional dependencies.

Calculating with FDs Let $\mathcal{F}$ denote the set of the functional dependencies expressed in natural language in statements of the exercise. The exercise essentially asks to decide for various $X \rightarrow Y$ whether $\mathcal{F}$ entails $X \rightarrow Y$. For this task there is a nice theorem, discussed in the lectures:

$$\mathcal{F} \text{ entails } X \rightarrow Y \quad \Leftrightarrow \quad Y \subseteq X^+_{\mathcal{F}}$$

For the computation of $X^+_{\mathcal{F}}$, being the *closure of X with respect to $\mathcal{F}$*, there exists a simple and efficient algorithm. We omit, here and in the sequel, the subscript $\mathcal{F}$ in the notation $X^+_{\mathcal{F}}$, and shall present the steps of the algorithm as a series of equations. For example:

$$T^+ \;=\; T\ldots \;\overset{(8)}{=}\; Tb\ldots \;\overset{(9)}{=}\; Tbk\ldots \;\overset{(2)}{=}\; TbkK\ldots$$

In the first step, T itself is asserted to be in T^+ (on account of the definition of closure), and in each following step the set is extended further by applying one of the members of $\mathcal{F}$, as indicated on top of the = sign. (For the formalists: an expression of the form '$X\ldots$' actually means 'X^+'.) In the example, we see after some steps that $K \in T^+$, so $T \rightarrow K$ is entailed by $\mathcal{F}$. When no given functional dependency produces a new attribute, the closure has been computed completely, which we indicate by omitting the ellipses ($\ldots$). Complete knowledge of the closure X^+ is needed to conclude $Y \not\subseteq X^+$, and hence $X \not\rightarrow Y$. Here is an example (from which $T \not\rightarrow P$ may be concluded):

$$T^+ \;= T\ldots \overset{(8)}{=} Tb\ldots \overset{(9)}{=} TbK\ldots \overset{(11')}{=} TbKR\ldots \;=\; TbKR$$

It is easy to verify that no FD will produce a new attribute from the extreme right hand side.

In this way denials of functional dependencies are not, and need not, used at all. However, we could use them as follows: if $X \not\rightarrow Y$ is given, and from $X' \rightarrow Y'$ we are able to derive $X \rightarrow Y$, then $X' \rightarrow Y'$ is not true. (Another usage would be to show that for one of the given denials we are also able to derive the confirmation (together an inconsistency), allowing us to conclude whatever we want! Fortunately, the given assertions are consistent, although I do not prove that here.)

Preparation for the exercise First, we observe that the exercise speaks about *bestelling* and its unique time stamp T; it may be concluded that T identifies *bestelling*en. However, we shall not use this identification (thus making it harder for ourselves), but instead use b as identifier for *bestelling*en, and we will find a relation between b and T from the case text (namely: $b \rightarrow T$ and $T \rightarrow b$). In order not to make it too hard, we *do* assume that *klant*en and *product*en are identified by K and P, respectively. (We could have assumed that there are identifiers k and p for *klant*en and *product*en, in which case we should have find relations between k, p, K, P from the text — which is quite well possible). Also, regarding *Rekeningnummer*s, *Hoeveelheden*, *Omschrijving*en, and *Beoordeling*en, we assume that these are identified by R, H, O, and B, respectively. Thus, we are dealing with a relation schema $Bestelling = (bTKRPHOB; \mathcal{F})$ with *virtual* attribute b, where $\mathcal{F}$ (and some more information) is given in natural language.

Second, we observe that in almost every sentence in the case text, there is a dependency on *bestelling* (for which we have introduced identifier b). This is further affirmed by the fragments:

> "Iedere rij in de tabel hoort bij één bestelling…", and
> "De interpretatie van de attributen is, per rij van de tabel, …"

So, whenever "bestelling" or "besteld" occurs in a sentence, identifier b will play a role in the formalization. (This explains a lot of student errors.)

Elaboration of the exercise The exercise essentially asks to decide for various $X \rightarrow Y$ whether it is entailed by $\mathcal{F}$. So we first express the given statement as functional dependencies or denials of functional dependencies. It turns out that some information is given twice or thrice; the repeated information is left unnumbered. The result is in Figure 3 on page 6.

Having done that, we can decide the truth of each proposed functional dependency following the method explained above. The result is in Figure 4 on page 7.

Conclusion Making errors is a human feature. The proposed approach, extended with machine assistance, minimizes the chance for errors, and thus helps to improve the quality of database design. Only the the given sentences have to be discussed with the principals; the consequences are correct by calculation! (But, of course, it is wise to do some checks on the consequences that have been calculated, be it only to detect errors in the given initial statements.)

As a demonstration of an error sneaked into my informal reasoning, I found only after completion of the formal calculations that in the initially proposed solution, my motivation "volgens S14" for $P \not\rightarrow T$ (in the one-but-last line of the table to be filled in) is not at all sufficient; sentence S14 reads "Een product kan door meerdere klanten besteld worden" meaning $bP \not\rightarrow K$, and some more statements (for example $T \rightarrow K$, derivable from S8 and S9) are needed to draw the desired conclusion.

Role of the theorem. The theorem that forms the basis of this approach is well known, but —as far as I know— *only* presented in the context of normalization (in order to decide whether an FD violates the BCNF condition, and to compute the FDs for the components of the decomposition). I am happy to observe that the theorem can also be used quite early in the stage of database design.

Formal methods. The proposed method favors formal methods over informal reasoning (in order to improve the quality of the results). As such it is in line with my earlier proposal about the construction of SQL queries: give a direct translation in set and logic notation of the natural language query, and then continue to transform by *formal calculation* the expression into a form that closely matches SQL. Since there exist formal rules for converting an ERD into a database schema, the *only* informal reasoning involved in an initial database design is thus the construction of an ERD. I wait for formal methods (apart from machine support) to assist in this activity.

Poor students. During the writing of this note, I've got inspiration for a whole bunch of new and interesting exam questions for the students...

The Exercise

Voor een bedrijf dat producten verkoopt aan klanten, wordt een database ontworpen waarin een tabel *Bestelling* voorkomt met de attributen T, K, R, P, H, O, B. Iedere rij in de tabel hoort bij één bestelling, maar bij een bestelling horen mogelijk meerdere rijen.
De interpretatie van de attributen is, per rij van de tabel, als volgt:

S1. T: het Tijdstip (*time stamp*) van de bestelling
S2. K: de Klantnaam van een klant die de bestelling doet
S3. R: het Rekeningnummer dat klant K bij de bestelling heeft opgegeven
S4. P: het Productnummer van een product besteld door klant K
S5. H: de Hoeveelheid waarin product P door klant K besteld is
S6. O: de Omschrijving van product P
S7. B: de Beoordeling van product P door klant K, indien bestaand

Toelichting:

S8. Verschillende bestellingen hebben verschillende tijdstippen (*time stamps*).
S9. Bij een bestelling is precies één klant betrokken.
S10. Verschillende klanten hebben verschillende klantnamen.
S11. Een klant kan verschillende rekeningnummers opgeven in verschillende bestellingen, maar niet in één bestelling.
S12. Verschillende producten hebben verschillende productnummers.
S13. Een klant kan een product in één bestelling hooguit éénmaal bestellen.
S14. Een product kan door meerdere klanten besteld worden.
S15. Een klant kan een product hooguit één beoordeling geven.
Heeft een klant een product niet beoordeeld, dan wordt er zo nodig een *default* beoordeling genomen voor de beoordeling van dat product door die klant.
S16. Een product kan door verschillende klanten beoordeeld worden.
S17. Een product heeft precies één omschrijving.
S18. Verder zijn er geen beperkingen aan de manier waarop de T, K, R, P, H, O, B-waarden van een rij uit de tabel aan elkaar gerelateerd zijn.

Geef in de tabel hieronder, bij iedere functionele afhankelijk, met de letters W, O aan of de functionele afhankelijk naar verwachting Waar of Onwaar is in de tabel *Bestelling*, en geef steeds een korte motivatie waarin u zo mogelijk de regelnummers $S1 \,..\, S18$ uit de opgave gebruikt:

FD	W/O	motivatie
$T \rightarrow K$		
$T \rightarrow R$		
$T \rightarrow P$		
$K \rightarrow R$		
$KT \rightarrow P$		
$KP \rightarrow B$		
$P \rightarrow O$		
$P \rightarrow K$		
$P \rightarrow T$		
$PB \rightarrow K$		

Figuur 1: The exercise

The exercise together with a Solution

Voor een bedrijf dat producten verkoopt aan klanten, wordt een database ontworpen waarin een tabel *Bestelling* voorkomt met de attributen T, K, R, P, H, O, B. Iedere rij in de tabel hoort bij één bestelling, maar bij een bestelling horen mogelijk meerdere rijen.
De interpretatie van de attributen is, per rij van de tabel, als volgt:

S1. T: het Tijdstip (*time stamp*) van de bestelling
S2. K: de Klantnaam van een klant die de bestelling doet
S3. R: het Rekeningnummer dat klant K bij de bestelling heeft opgegeven
S4. P: het Productnummer van een product besteld door klant K
S5. H: de Hoeveelheid waarin product P door klant K besteld is
S6. O: de Omschrijving van product P
S7. B: de Beoordeling van product P door klant K, indien bestaand

Toelichting:

S8. Verschillende bestellingen hebben verschillende tijdstippen (*time stamps*).
S9. Bij een bestelling is precies één klant betrokken.
S10. Verschillende klanten hebben verschillende klantnamen.
S11. Een klant kan verschillende rekeningnummers opgeven in verschillende bestellingen, maar niet in één bestelling.
S12. Verschillende producten hebben verschillende productnummers.
S13. Een klant kan een product in één bestelling hooguit éénmaal bestellen.
S14. Een product kan door meerdere klanten besteld worden.
S15. Een klant kan een product hooguit één beoordeling geven.
Heeft een klant een product niet beoordeeld, dan wordt er zo nodig een *default* beoordeling genomen voor de beoordeling van dat product door die klant.
S16. Een product kan door verschillende klanten beoordeeld worden.
S17. Een product heeft precies één omschrijving.
S18. Verder zijn er geen beperkingen aan de manier waarop de T, K, R, P, H, O, B-waarden van een rij uit de tabel aan elkaar gerelateerd zijn.

Geef in de tabel hieronder, bij iedere functionele afhankelijk, met de letters W, O aan of de functionele afhankelijk naar verwachting Waar of Onwaar is in de tabel *Bestelling*, en geef steeds een korte motivatie waarin u zo mogelijk de regelnummers $S1 \,..\, S18$ uit de opgave gebruikt:

FD	W/O	motivatie
		T identificeert bestellingen volgens S8 en S1 K identificeert klanten volgens S10 en S2 P identificeert producten volgens S12 en S4
$T \rightarrow K$	W	volgens S9
$T \rightarrow R$	W	volgens S9, S11
$T \rightarrow P$	O	ondanks S13; want een K kan per T meerdere P bestellen!
$K \rightarrow R$	O	volgens S11
$KT \rightarrow P$	O	ondanks S13; een K kan per T meerdere P bestellen!
$KP \rightarrow B$	W	volgens S15 (NB: S13 is foute motivatie!)
$P \rightarrow O$	W	volgens S17
$P \rightarrow K$	O	volgens S14
$P \rightarrow T$	O	volgens S14
$PB \rightarrow K$	O	volgens S16 (NB: S15 is foute motivatie!)

Figuur 2: The proposed solution

Formal Solution, part 1

n	sentence n (with klant=klantnaam, product=productnummer)	our conclusion	
S1	T: *het* tijdstip van *de* bestelling	$b \to T$	(1)
S2	K: een klant die de bestelling doet	⟨not an FD⟩	
S3	R: het rekeningnummer dat klant K bij *de* bestelling heeft opgegeven	$bK \to R$	—
S4	P: een product besteld door klant K	⟨not an FD⟩	
S5	H: de hoeveelheid waarin product P door klant K *besteld* is	$bKP \to H$	—
S6	O: de omschrijving van product P	$P \to O$	—
S7	B: de beoordeling van product P door klant K, indien bestaand.	$KP \to B$	—
S8	Verschillende bestellingen hebben verschillende tijdstippen.	$T \to b$	(8)
S9	Bij een bestelling is precies één klant betrokken.	$b \to K$	(9)
S10	Verschillende klanten hebben verschillende klantnamen.	K identifies *klant*	
S11	Een klant kan verschillende rekeningnummers opgeven in verschillende bestellingen, maar niet in één bestelling.	$K \not\to R$ $bK \to R$	(11) (11′)
S12	Verschillende producten hebben verschillende productnummers.	P identifies *product*	
S13	Een klant kan een product in één bestelling hooguit éénmaal bestellen. *Here some interpretation was needed to formalise the phrase "hooguit éénmaal bestellen".*	$bKP \to H$	(13)
S14	Een product kan door meerdere klanten besteld worden.	$bP \not\to K$	(14)
S15	Een klant kan een product hooguit één beoordeling geven. Heeft een klant een product niet beoordeeld, dan wordt er zo nodig een *default* beoordeling genomen voor de beoordeling van dat product door die klant.	$KP \to B$ ⟨not an FD⟩	(15)
S16	Een product kan door verschillende klanten beoordeeld worden.	$PB \not\to K$	(16)
S17	Een product heeft precies één omschrijving.	$P \to O$	(17)
S18	Verder zijn er geen beperkingen aan de manier waarop de T, K, R, P, H, O, B-waarden van een rij uit de tabel aan elkaar gerelateerd zijn.	⟨not an FD⟩	

Figuur 3: Formal solution, part 1.
The unnumbered FDs occur, *with* a number, further down in the table.

Formal Solution, part 2

$T \to K$	W	$T^+ = T\ldots \overset{(8)}{=} Tb\ldots \overset{(9)}{=} TbK\ldots$
$T \to R$	W	$T^+ = T\ldots \overset{(8)}{=} Tb\ldots \overset{(9)}{=} TbK\ldots \overset{(11')}{=} TbKR\ldots$
$T \to P$	O	$T^+ = T\ldots \overset{(8)}{=} Tb\ldots \overset{(9)}{=} TbK\ldots \overset{(11')}{=} TbKR\ldots = TbKR$
$K \to R$	O	$K^+ = K\ldots = K$ (also immediate from (11): $K \not\to R$)
$KT \to P$	O	$(KT)^+ = KT\ldots \overset{(8)}{=} KTb\ldots \overset{(3)}{=} KTbR\ldots = KTbR$
$KP \to B$	W	$(KP)^+ = KP\ldots \overset{(15)}{=} KPB\ldots$
$P \to O$	W	$P^+ = P \overset{(17)}{=} PO\ldots$
$P \to K$	O	$P^+ = P \overset{(6)}{=} PO\ldots = PO$ (also immediate from (14): $bP \not\to K$)
$P \to T$	O	$P^+ = PO$ see previous line (also: $T \to K$ and $P \not\to K \Rightarrow P \not\to K$)
$PB \to K$	O	$(PB)^+ = PB \overset{(6)}{=} PBO\ldots = PBO$ (also from (16): $PB \not\to K$)

Clearly, first doing the completed computation for T^+ (as in the 3rd line), allows us to short-cut the computation in the 1st and 2nd line. Also, the last line implies the 2nd-but-last line, without further computation.

Figuur 4: Formal solution, part 2