Chapter 15: Basics of Functional Dependencies and Normalization for Relational Databases

De: Elmasri et Navathe, Fundamentals of Database Systems
Traduction: Celine Kuttler

CHAPTER 15: BASICS of FUNCTIONAL DEPENDENCIES AND NORMALIZATION FOR RELATIONAL DATABASES

Exo 1 (15.27)

Considerez une relation *R(A,B,C,D,E)* avec les dépendances suivantes:

DF1: *AB → C*
DF2: *CD → E*
DF3: *DE → B*
AB est elle clé candidate de cette relation? Si non, est ce le cas de ABC ? Justifiez vos réponses.
Answers:
Non, AB+ = {A,B,C},est un vrai sous-ensemble de {A,B,C,D,E}
Oui, ABD+ = {A,B,C,D,E}

Details :
Developpement de AB+ :
{AB}
DF1 → {ABC}
saturation, puisqu'aucune autre DF n'est applicable. Pour cela il faudrait trouver tous les elements de la partie droite d'une DF.

Developpement de ABD+ :
{ABD}
DF1→ {ABCD}
DF2 → {ABCDE}
saturation !

Exo 2

Les cles candidates sont A, E, CD et BC. On le voit en calculant les clotures A+, E+, CD+ et BC+ et relisant bien les définitions.

Le probleme en cherchant toutes les clés candidates, est qu'il faut potentiellement tester tous les sous-ensemble des attributs du schéma, et qu'il y a un nombre exponentiel de tels sous-ensembles.

On peut également utiliser l'outil pour obtenir toutes les clés candiates.

```
schema([a,b,c,d,e]).

fds([
[[a],[b,c]],
[[c,d],[e]],
[[b],[d]],
[[e],[a]]
]
```

```
).

%schema(R),fds(F),candkey(R,F,K).

%?- schema(R),fds(F),candkey(R,F,K).
# R = [a, b, c, d, e],
# F = [[[a], [b, c]], [[c, d], [e]], [[b], [d]], [[e], [a]]],
# K = [e] ;
# R = [a, b, c, d, e],
# F = [[[a], [b, c]], [[c, d], [e]], [[b], [d]], [[e], [a]]],
# K = [c, d] ;
# R = [a, b, c, d, e],
# F = [[[a], [b, c]], [[c, d], [e]], [[b], [d]], [[e], [a]]],
# K = [b, c] ;
# R = [a, b, c, d, e],
# F = [[[a], [b, c]], [[c, d], [e]], [[b], [d]], [[e], [a]]],
# K = [a] ;
# false.

# response:
# les cles candidates sont A, E, CD et BC
```

Tableau 1 : Exo 2 avec DBD.

Exo 3 :

Q1 : Montrer A → C
L'extension de A → B avec A donne : AA → AB . Or AA = A. Donc on a A → AB. Avec B → C, qui était donné, on obtient par transitivité A → C.

Q2 :Peut-on déduire B → C ?
Non. Il faut avoir A ET B pour connaître la valeur de C.
Prenez comme exemple la table de la boutique, Catalogue(aid,fid, prix), ou A:aid, B:fid, C:prix. De penser B → C reviendrait a penser que tous les articles d'un vendeur sont offerts au meme prix.

Pour une preuve complete, il faudrait calculer la cloture de l'ensemble de Dfs et puis constater que B → C n'y est pas. On peut faire faire ce test par l'outil. Soit en faisant afficher le total de la cloture des DFs **(avec fplus)** , soit utilisant le predicat **finfplus** dédié au test d'appartenance d'une DF dans la cloture d'un schéma avec DFs.

```
schema([a,b,c]).

fds([
[[a],[b]],
[[a,b],[c]]
]
```

```
).

% Q1
%schema(R),fds(F),finfplus(R,F,[[a],[c]]).
% YES et pas trivial a voir.

%Q2
%schema(R),fds(F),finfplus(R,F,[[b],[c]]).
% false
```

Tableau 2 : Exo 3.

Exo 4 (15.24)

Prenez en compte la relation universelle R = {A, B, C, D, E, F, G, H, I, J} avec l'ensemble de dépendances fonctionnnelles F = { {A, B} -> {C}, {A} -> {D, E}, {B} -> {F}, {F} -> {G, H}, {D} -> {I, J} }.

- Quelle est la clé de R ?
- Décomposez R en 2NF, puis en relations en 3NF .

Corrigé:
Un ensemble minimal d'attributs dont la clôture comprend tous les attributs de R est une clé (il est possible d'appliquer l'algo 15.4a, voir chapitre 15 du livre).

Puisque la clôture de {A, B} est **{A, B}+ = R**, une des clés de R est {A, B} (et dans ce cas, il s'agit de la seule clé).

Pour normaliser R de manière intuitive en 2NF, et ensuite en 3NF, nous passons par les étapes suivantes (alternativement, il est possible d'appliquer les algorithmes du chapitre 15).

En premier, il faut identifier les dépendances partielles qui violent la 2NF. Il s'agit d'attibuts qui sont partiellement dépendants d'une partie de la clé sans l'autre, donc ici, {A} ou {B} tout seul. Nous calculons les clôtures {A}+ et {B}+ afin de trouver des attributs partiellement dépendants.

{A}+ = {A, D, E, I, J}.
Donc {A} -> {D, E, I, J} (nous ne prenons pas en compte la dépendance triviale {A} -> {A})

{B}+ = {B, F, G, H},
donc {A} -> {F, G, H} (ici encore nous ne prenons pas en compte la dépendance triviale {B} -> {B})

Pour normaliser en 2NF, nous retirons de R les attributs qui dépendent d'une partie de la clé (A ou B) et les mettons dans deux nouvelles relations séparées R1 et R2, avec la partie de la clé de laquelle ils dépendent (A ou B). Ces derniers sont copiés dans les nouvelles relations, mais restent également dans la relation originale, que nous renommons R3 dans le suivant :
R1 = {<u>A</u>, D, E, I, J}, R2 = {<u>B</u>, F, G, H}, R3 = {<u>A, B</u>, C}
Les nouvelles clés pour R1, R2 et R3 sont souligées.

Ensuite, nous cherchons des dépendances transitives pour R1, R2 et R3, qui violent la 3NF.

La relation R1 a la dependance transitive {A} -> {D} -> {I, J},
donc nous mettons les attributs dependants transitivement {I,J} de R11, dans une nouvelle relation R11, et nous copions l'attribut D duquel ils dépendent dans R11. Les autres attributs restent dans une nouvelle relation R12.
Donc R1 est décomposée en R11 et R12 comme ceci :
R11 = {D, I, J}, R12 = {A, D, E}

De maniere similaire la relation R2 est decomposée en R21 et R22 grâce à la dependance transitive {B} -> {F} -> {G, H}:

R21 = {F, G, H}, R22 = {B, F}

L'ensemble final de relations en troisième forme normale est {R11, R12, R21, R22, R3}

Exo 5 15.21

Dans quelle forme normale est la relation LOTS, par rapport aux interprétations de formes normales restrictives, qui ne prennent en compte que les clés primaires?

Est-ce la même forme normale qu'en prenant en compte les définitions plus générales?

Réponse:

Si nous ne prenons que en compte la clé primaire, la relation LOTS de la figure est en 2NF puisqu'il n'y a pas de dependances partielles de la clé primaire.

Mais, elle n'est pas en 3NF, puisqu'il y a deux dépendances transitives à partir de la clé primaire:
PROPERTY_ID# ->COUNTY_NAME ->TAX_RATE, et
PROPERTY_ID# ->AREA ->PRICE.

Si nous prenons en compte toutes les clés et utilisons les définitions générales de 2NF et 3NF, la relation LOTS ne sera qu'en premiere forme normale. La raison est la dépendence partielle
COUNTY_NAME ->TAX_RATE
qui dépend de la clé secondaire {COUNTY_NAME, LOT#}, ce qui viole la 2NF.

Exo 6 (15.29)

Analysez les relations suivantes, pour l'application avec laquelle la société ABC gère des commandes.

COMMANDE(C#,Cdate, Client#,Somme_totale)
COMMANDE_PIECE(C#,P#,Qte_cde,Prix_total,Remise%)

Supposez que chaque pièce a une remise différente des autres. Le *Prix_total* est relatif à une pièce, *Cdate* est le jour ou commande a été passée, et *Somme_totale* la somme de la facture pour cette commande.

Q1 Si nous faisons une jointure naturelle entre les deux relations COMMANDE et COMMANDE_PIECE de cette base, quel sera le schéma qui en résulte ?
Q2 Quelle sera la clé ?
Q3 Indiquez les dépendances fonctionnelles de cette nouvelle relation.
Q4 Est elle en seconde forme normale? Est elle en 3NF? Pourquoi ou pourquoi non? Indiquez toutes les hypothèses que vous devez supposer.

Réponse:

Etant donné les relations
Commande(C#, Cdate, Client#, Somme_totale)
Commande_Piece(C#, P#, Qte_cde, Prix_total, Remise%),

le schéma du produit naturel
Commande * Commande_Piece est le suivant

T_1(C#,P#,Cdate, Client#, Somme_totale, Qte_cde, Prix_total, Remise%)
et sa clé est C#,P#.

On a les dependances fonctionnelles
C#P# → Qte_cde
C#P# → Prix_total
C#P# → Remise%
C# → Cdate
C# → Client#
C# → Somme_totale
Ce schéma n'est pas en 2NF, puisque les attribtuts Cdate, Client#, et Somme_totale ne dépendent que partiellement de la clé primaire C#P#.
Le schéma n'est pas en 3NF non plus, puisque pour cela la 2NF est un pré-requis.

Exo 7 15.31

Voila une relation pour des livres qui sont publiés:
LIVRE(Livre_titre, Nomauteur, Livre_type, Prix_liste, Auteur_afil, Editeur)
L'attribut *Auteur_afil* indique l'affiliation d'un auteur, p.ex. son université. Supposez les dépendances suivantes:
Livre_titre → Editeur, Livre_type ,
Livre_type → Prix_liste ,
Nomauteur → Auteur_afil
(a) Dans quelle forme normale est cette relation?
(b) Normalisez en relations plus petites, jusqu'à ce qu'il n'y ait plus de décompositions supplémentaires possibles. Indiquez les raisons pour chaque étape de décomposition.

Réponse:

Etant donné la relation
Livre(Livre_titre, Nomauteur, Livre_type, Prix_liste, Auteur_afil, Editeur)
et les DFs
Livre_titre → Editeur, Livre_type

Livre_type → Prix_liste
Nomauteur → Auteur_afil

(a)La clé de cette relation est
Livre_titre,Nomauteur.
Cette relation est en 1NF (rien n'est imbriqué). Mais elle n'est pas en 2NF, puisqu'il n'y a aucun attribut qui dépend completement fonctionnellement de la clé. Elle n'est donc pas en 3NF non plus.

(b) décomposition en 2NF :
Livre0(Livre_titre, Nomauteur)
Livre1(Livre_titre, Editeur, Livre_type, Prix_liste)
Livre2(Nomauteur, Auteur_afil)

Cette décomposition élimine les dépendances partielles.

Décomposition en 3NF :
Livre0(Livre_titre, Nomauteur)
Livre1-1(Livre_titre, Editeur, Livre_type)
Livre1-2(Livre_type, Prix_liste)
Livre2(Nomauteur, Auteur_afil)

Cette décomposition élimine la dépendance transitive de Prix_liste.