Théorie de la normalisation relationnelle (dépendance fonctionnelle, forme normale, clé)

Table des matières

Cours

La théorie de la normalisation relationnelle est très importante pour la conception de *BD*∗, dans la mesure où elle donne le cadre théorique pour la gestion de la redondance, et dans la mesure où une bonne maîtrise de la redondance est un aspect majeur de cette conception.

1. Redondance et normalisation

Objectifs

Comprendre la problématique de la redondance.

1.1. Exercice : Introduction à la redondance

Soit la relation R suivante, définie en extension :

A	B	C	D	E	F	G
0	1	1	10	5	X	A
0	2	1	10	9	X	G
0	1	2	10	6	X	S
0	1	3	10	7	X	D
1	2	3	20	7	Y	D
0	3	3	10	9	X	G
1	4	3	20	8	Y	F
1	1	4	20	9	Y	G

Relation R

Question 1

Proposez des clés pour cette relation. Justifiez brièvement.

Question 2

Cette relation contient-elle des redondances ? Si oui lesquelles ? Justifiez brièvement.

Question 3

Si la relation contient des redondances, proposez une solution contenant exactement la même information, mais sans redondance.

1.2. Les problèmes soulevés par une mauvaise modélisation

Attention

Il y a toujours plusieurs façons de modéliser conceptuellement un problème, certaines sont bonnes et d'autres mauvaises. C'est l'expertise de l'ingénieur en charge de la modélisation, à travers son expérience accumulée et sa capacité à traduire le problème posé, qui permet d'obtenir de bons modèles conceptuels.

S'il est difficile de définir un bon modèle conceptuel, on peut en revanche poser qu'un bon modèle logique relationnel est un modèle où la redondance est contrôlée.

On peut alors poser qu'un bon modèle conceptuel est un modèle conceptuel qui conduit à un bon modèle relationnel, après application des règles de passage E-A ou UML vers relationnel. Mais on ne sait pas pour autant le critiquer avant ce passage, autrement qu'à travers l'œil d'un expert.

A défaut de disposer d'outils systématiques pour obtenir de bons modèles conceptuels, on cherche donc à critiquer les modèles relationnels obtenus.

La théorie de la normalisation est une théorie qui permet de critiquer, puis d'optimiser, des modèles relationnels, de façon à en contrôler la redondance.

Un mauvais modèle relationnel

Exemple

Imaginons que nous souhaitions représenter des personnes, identifiées par leur numéro de sécurité sociale, caractérisées par leur nom, leur prénom, ainsi que les véhicule qu'elles ont acheté, pour un certain prix et à une certaine date, sachant qu'un véhicule est caractérisé par son numéro d'immatriculation, un type, une marque et une puissance. On peut aboutir à la représentation relationnelle suivante :

```
Personne(NSS, Nom, Prénom, Immat, Marque, Type, Puiss, Date, Prix)
```

Posons que cette relation soit remplie par les données suivantes :

NSS	Nom	Prénom	Immat	Marque	Type	Puiss	Date	Prix
16607...	Dupont	Paul	AJ600AQ	Renault	Clio	5	1/1/96	60000
16607...	Dupont	Paul	AA751KK	Peugeot	504	7	2/7/75	47300
24908...	Martin	Marie	AA751KK	Peugeot	504	7	1/10/89	54900
15405...	Durand	Olivier	AA751KK	Peugeot	504	7	8/8/90	12000
15405...	Durand	Olivier	AJ600AQ	Renault	Clio	5	7/6/98	65000
15405...	Durand	Olivier	XX100XX	BMW	520	10	4/5/01	98000

Relation redondante

On peut alors se rendre compte que des redondances sont présentes, si l'on connaît NSS on connaît Nom et Prénom, si on connaît Immat, on connaît Marque, Type et Puiss.

NSS	Nom	Prénom	Immat	Marque	Type	Puiss	Date	Prix
16607...	Dupont	Paul	AJ600AQ	Renault	Clio	5	1/1/96	60000
16607...	Dupont	Paul	AA751KK	Peugeot	504	7	2/7/75	47300
24908...	Martin	Marie	AA751KK	Peugeot	504	7	1/10/89	54900
15405...	Durand	Olivier	AA751KK	Peugeot	504	7	8/8/90	12000
15405...	Durand	Olivier	AJ600AQ	Renault	Clio	5	7/6/98	65000
15405...	Durand	Olivier	XX100XX	BMW	520	10	4/5/01	98000

Relation redondante

On sait que ces redondances conduiront à des problèmes de contrôle de la cohérence de l'information (erreur dans la saisie d'un numéro de sécurité sociale), de mise à jour (changement de nom à reporter dans de multiples tuples), de perte d'information lors de la suppression de données (disparition des informations concernant un type de véhicule) et de difficulté à représenter certaines informations (un type de véhicule sans propriétaire).

Complément

On conseillera de lire le chapitre 2 de *SQL2 SQL3, applications à Oracle*∗ (pages 42 à 49) qui propose une très bonne démonstration par l'exemple des problèmes posés par une mauvaise modélisation relationnelle.

1.3. Principes de la normalisation

Fondamental

La théorie de la normalisation est une théorie destinée à concevoir un bon schéma d'une base de données sans redondance d'information et sans risques d'anomalie de mise à jour. Elle a été introduite dès l'origine dans le modèle relationnel.

La théorie de la normalisation est fondée sur deux concepts principaux :

- **Les dépendances fonctionnelles**

  Elles traduisent des contraintes sur les données.
- **Les formes normales**

  Elles définissent des relations bien conçues.

La mise en œuvre de la normalisation est fondée sur la décomposition progressive des relations jusqu'à obtenir des relations normalisées.

Méthode

Afin de mener une bonne conception on cherchera à obtenir un modèle relationnel en *BCNF*∗ pour lequel l'absence de redondance est simple à démontrer (car les *DF*∗ sont simples à vérifier).

2. Dépendances fonctionnelles

Objectifs

Savoir repérer et exprimer des dépendances fonctionnelles.

Définir une clé par les dépendances fonctionnelles.

2.1. Exercice : A1, dans l'eau !

Considérons le schéma de la relation suivante :

- `r (A, B, C, D, E)`

Cette relation est définie en extension par les tuples suivants :

A	B	C	D	E
a1	b2	c2	d3	e2
a1	b2	c2	d1	e4
a2	b3	c2	d1	e5
a2	b4	c5	d1	e5

Parmi les dépendances fonctionnelles suivantes, lesquelles s'appliquent à r ?

- [ ] E→D
- [ ] D→E
- [ ] C→A
- [ ] E→B

- [ ] E→A
- [ ] B→C
- [ ] B→D
- [ ] B→A

2.2. Dépendance fonctionnelle

Dépendance fonctionnelle ***Définition***

Soient R(A1, A2, ... , An) un schéma de relation, X et Y des sous-ensembles de A1, A2, ... , An. On dit que X détermine Y, ou que Y dépend fonctionnellement de X, si et seulement s'il existe une fonction qui à partir de toute valeur de X détermine une valeur unique de Y.

Plus formellement on pose que X détermine Y pour une relation R si et seulement si quelle que soit l'instance r de R, alors pour tous tuples t1 et t2 de r on a :

Projection (t1,X) = Projection (t2,X) ⇒ Projection (t1,Y) = Projection (t2,Y)

Syntaxe

Si X détermine Y, on note : X→Y

Exemple

Soit la relation R suivante :

```
Personne(NSS, Nom, Prénom, Marque, Type, Puiss, Date, Prix)
```

On peut poser les exemples de *DF*∗ suivants :

- NSS→Nom
- NSS→Prénom
- Type→Marque
- Type→Puiss
- (NSS, Type, Date)→Prix
- etc.

Comment trouver les DF ? ***Remarque***

Une DF est définie sur l'intension du schéma et non son extension. Une DF traduit une certaine perception de la réalité. Ainsi la DF (NSS, Type, Date)→Prix signifie que personne n'achète deux voitures du même type à la même date.

La seule manière de déterminer une DF est donc de regarder soigneusement ce que signifient les attributs et de trouver les contraintes qui les lient dans le monde réel.

Pourquoi trouver les DF ? ***Remarque***

Les DF font partie du schéma d'une BD, en conséquence, elles doivent être déclarées par les administrateurs de la BD et être contrôlées par le SGBD.

De plus l'identification des DF est la base indispensable pour déterminer dans quelle forme normale est une relation et comment en diminuer la redondance.

2.3. Axiomes d'Armstrong

Introduction

Les *DF*∗ obéissent à des propriétés mathématiques particulières, dites axiomes d'Armstrong.

Réflexivité ***Définition***

Tout groupe d'attributs se détermine lui même et détermine chacun de ses attributs (ou sous groupe de ses attributs).

Soient X et Y des attributs :

XY→XY et XY→X et XY→Y

Augmentation ***Définition***

Si un attribut X détermine un attribut Y, alors tout groupe composé de X enrichi avec d'autres attributs détermine un groupe composé de Y et enrichi des mêmes autres attributs.

Soient X, Y et Z des attributs :

X→Y ⇒ XZ→YZ

Transitivité ***Définition***

Si un attribut X détermine un attribut Y et que cet attribut Y détermine un autre attribut Z, alors X détermine Z.

Soient X, Y et Z des attributs :

X→Y et Y→Z ⇒ X→Z

2.4. Autres propriétés déduites des axiomes d'Armstrong

Introduction

A partir des axiomes d'Amstrong, on peut déduire un certain nombre de propriétés supplémentaires.

Pseudo-transitivité ***Définition***

Si un attribut X détermine un autre attribut Y, et que Y appartient à un groupe G qui détermine un troisième attribut Z, alors le groupe G' obtenu en substituant Y par X dans G détermine également Z.

Soient, W, X, Y et Z des attributs :

X→Y et WY→Z ⇒ WX→Z

Cette propriété est déduite de l'augmentation et de la transitivité :

X→Y et WY→Z ⇒ WX→WY et WY→Z ⇒ WX→Z

Union ***Définition***

Si un attribut détermine plusieurs autres attributs, alors il détermine tout groupe composé de ces attributs.

Soient X, Y et Z des attributs :

X→Y et X→Z ⇒ X→YZ

Cette propriété est déduite de la réflexivité, de l'augmentation et de la transitivité :

X→Y et X→Z ⇒ X→XX et XX→XY et YX→YZ ⇒ X→YZ

Décomposition *Définition*

Si un attribut détermine un groupe d'attribut, alors il détermine chacun des attributs de ce groupe pris individuellement.

Soient X, Y et Z des attributs :

X→YZ ⇒ X→Z et X→Y

Cette propriété est déduite de la réflexivité et de la transitivité :

X→YZ ⇒ X→YZ et YZ→Z ⇒ X→Z

2.5. DF élémentaire

Dépendance fonctionnelle élémentaire *Définition*

Soit G un groupe d'attributs et A un attribut, une *DF*∗ G→A est élémentaire si A n'est pas incluse dans G et s'il n'existe pas d'attribut A' de G qui détermine A.

DF élémentaires *Exemple*

- AB→C est élémentaire si ni A, ni B pris individuellement ne déterminent C.
- Nom, DateNaissance, LieuNaissance→Prénom est élémentaire.

DF non élémentaires *Exemple*

- AB→A n'est pas élémentaire car A est incluse dans AB.
- AB→CB n'est pas élémentaire car CB n'est pas un attribut, mais un groupe d'attributs.
- N°SS→Nom, Prénom n'est pas élémentaire.

Remarque

On peut toujours réécrire un ensemble de DF en un ensemble de *DFE*∗, en supprimant les DF triviales obtenues par réflexivité et en décomposant les DF à partie droite non atomique en plusieurs DFE.

Réécriture de DF en DFE *Exemple*

On peut réécrire les DF non élémentaires de l'exemple précédent en les décomposant DFE :

- AB→A n'est pas considérée car c'est une DF triviale obtenu par réflexivité.
- AB→CB est décomposée en AB→C et AB→B, et AB→B n'est plus considérée car triviale.
- N°SS→Nom, Prénom est décomposée en N°SS→Nom et N°SS→Prénom.

2.6. Fermeture transitive des DFE

Fermeture transitive *Définition*

On appelle fermeture transitive F+ d'un ensemble F de *DFE*∗, l'ensemble de toutes les DFE qui peuvent être composées par transitivité à partir des DFE de F.

Exemple

Soit l'ensemble F = {A→B, B→C, B→D, A→E}.

La fermeture transitive de F est F+ = { A→B, B→C, B→D, A→E, A→C, A→D }

2.7. Couverture minimale des DFE

Couverture minimale — *Définition*

La couverture minimale d'un ensemble de *DFE*∗ est un sous-ensemble minimum des DFE permettant de générer toutes les autres DFE.

Synonymes : Famille génératrice

Remarque

Tout ensemble de DFE (et donc tout ensemble de DF) admet au moins une couverture minimale (et en pratique souvent plusieurs).

Exemple

L'ensemble F = {A→B, A→C, B→C, C→B} admet les deux couvertures minimales :

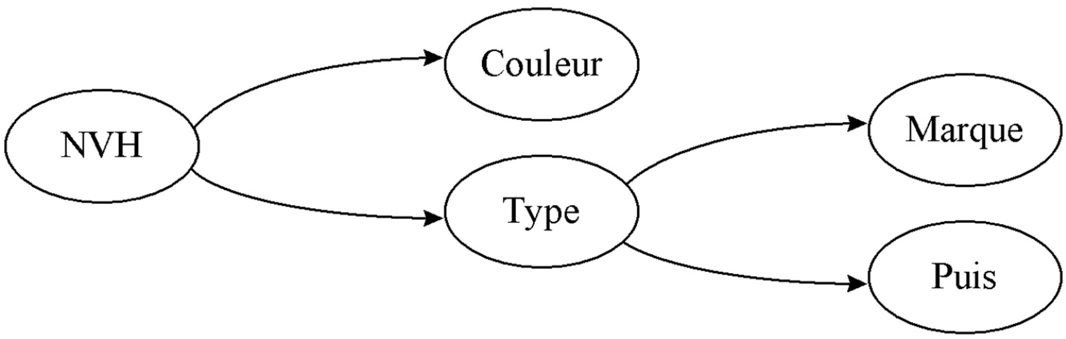
CM1 = {A→C, B→C, C→B} et CM2 = {A→B, B→C, C→B}

2.8. Graphe des DFE

On peut représenter un ensemble de DFE par un graphe orienté (ou plus précisément un réseau de Pétri), tel que les nœuds sont les attributs et les arcs les DFE (avec un seul attribut en destination de chaque arc et éventuellement plusieurs en source).

Relation Voiture — *Exemple*

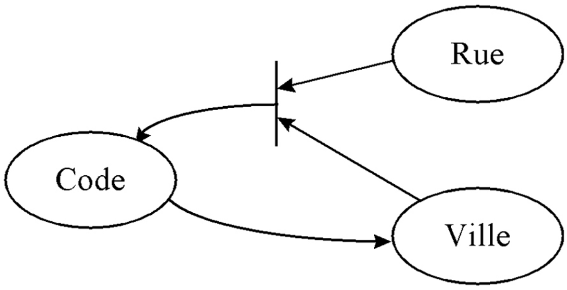
Soit la relation Voiture(NVH, Marque, Type, Puis, Couleur) avec l'ensemble des DF F = {NVH→Type, Type→Marque, Type→Puis, NVH→Couleur}. On peut représenter F par le graphe ci-dessous :



Graphe des DFE de la relation Voiture

Relation CodePostal — *Exemple*

Soit la relation CodePostal(Code, Ville, Rue) avec l'ensemble des DF F={Code→Ville, (Ville,Rue)→Code}. On peut réprésenter F par le graphe ci-dessous :



Graphe des DFE de la relation CodePostal

2.9. Définition formelle d'une clé

Clé — *Définition*

Soient une relation R(A1,A2,...,An) et K un sous-ensemble de A1,A2,... ,An.

K est une clé de R si et seulement si :

1. K→A1,A2,...,An
2. et il n'existe pas X inclus dans K tel que X→A1,A2,...,An.

Fondamental

Une clé est donc un ensemble minimum d'attributs d'une relation qui détermine tous les autres.

Clés candidates et clé primaire — *Remarque*

Si une relation comporte plusieurs clés, chacune est dite clé candidate et l'on en choisit une en particulier pour être la clé primaire.

Les clés candidates sont des clés ! — *Attention*

Toutes les clés candidates sont des clés, pas seulement la clé primaire.

Les clés candidates se déterminent mutuellement — *Remarque*

Toute clé candidate détermine les autres clés candidates, puisque qu'une clé détermine tous les attributs de la relation.

Relation "toute clé" — *Complément*

Étant donné qu'une relation dispose forcément d'une clé, si une relation R n'admet aucune clé K sous ensemble des attributs A1..An de R, alors c'est que K=A1..An (la clé est composée de **tous** les attributs de R).

On parle de relation "toute clé".

2.10. Exercice : A1, touché !

Considérons le schéma de la relation suivante :

- `r (A, B, C, D, E)`

Cette relation est définie en extension par les tuples suivants :

A	B	C	D	E
a1	b2	c2	d3	e2
a1	b2	c2	d1	e4
a2	b3	c2	d1	e5
a2	b4	c5	d1	e5

Après avoir énoncé les DF, déterminer, parmi les groupes d'attributs suivants, lesquels sont des clés ?

- [ ] A
- [ ] B
- [ ] C
- [ ] D
- [ ] E
- [ ] {B,E}
- [ ] {A,B,C,D,E}

3. Formes normales

Objectifs

Connaître les formes normales et leurs implications en terme de redondance.

3.1. Les formes normales

Les formes normales ont pour objectif de définir la décomposition des schémas relationnels, tout en préservant les *DF*∗ et sans perdre d'informations, afin de représenter les objets et associations canoniques du monde réel de façon non redondante.

On peut recenser les 6 formes normales suivantes, de moins en moins redondantes :

- la première forme normale
- la deuxième forme normale
- la troisième forme normale
- la forme normale de Boyce-Codd
- la quatrième forme normale
- la cinquième forme normale

La troisième forme normale est généralement reconnue comme la plus importante à respecter.

La *BCNF*∗ est la plus simple à établir.

3.2. Principe de la décomposition

L'objectif de la décomposition est de "casser" une relation en relations plus petites afin d'en éliminer les redondances et sans perdre d'information.

Décomposition *Définition*

La décomposition d'un schéma de relation R(A1,A2,...,An) est le processus de remplacement de ce schéma par une collection de schémas R1,R2,...,Rn telle qu'il est possible de reconstruire R par des opérations relationnelles de jointure sur R1,R2,...,Rn.

Décomposition préservant les DF *Définition*

Une décomposition d'une relation R en relations R1,R2,...Rn préserve les *DF*∗ si la fermeture transitive F+ des DF de R est la même que la fermeture transitive F+ de l'union des DF de R1,R2,...,Rn.

Décomposition préservant les DF d'une relation Voiture *Exemple*

Soit la relation Voiture(Numéro,Marque,Type,Puissance,Couleur) avec la fermeture transitive suivante :

- Numéro→Marque
- Numéro→Type
- Numéro→Puissance
- Numéro→Couleur
- Type→Marque
- Type→Puissance

On peut décomposer Voiture en préservant les DF en deux relations R1(Numéro,Type,Couleur) et R2(Type,Puissance,Marque).

3.3. Première forme normale

1NF *Définition*

Une relation est en *1NF*∗ si elle possède au moins une clé et si tous ses attributs sont atomiques.

Attribut atomique *Définition*

Un attribut est atomique si il ne contient qu'une seule valeur pour un tuple donné, et donc s'il ne regroupe pas un ensemble de plusieurs valeurs.

Avoir plusieurs métiers *Exemple*

Soit la relation Personne instanciée par deux tuples :

```
Personne(#Nom, Profession)
```

```
(Dupont, Géomètre)
(Durand, Ingénieur-Professeur)
```

La relation n'est pas en 1NF, car l'attribut Profession peut contenir plusieurs valeurs.

Pour que la relation soit en 1NF, on pourrait par exemple ajouter Profession à la clé et faire apparaître deux tuples pour Durand, on obtiendrait :

```
Personne(#Nom, #Profession)
```

```
(Dupont, Géomètre)
(Durand, Ingénieur)
(Durand, Professeur)
```

Une autre solution aurait été d'ajouter un attribut ProfessionSecondaire. On obtiendrait ainsi :

```
Personne(#Nom, Profession, ProfessionSecondaire)
```

```
(Dupont, Géomètre, Null)
(Durand, Ingénieur, Professeur)
```

Relativité de la notion d'atomicité *Remarque*

L'atomicité d'un attribut est souvent relative : on peut décider qu'un attribut contenant une date n'est pas atomique (et que le jour, le mois et l'année constituent chacun une valeur), ou bien que l'attribut est de domaine date et donc qu'il est atomique.

Énoncer les clés *Fondamental*

Le modèle relationnel impose qu'une relation ait une clé, donc la condition "est en 1NF si elle possède une clé" est superflue (au pire la relation est *toute clé*∗).

Il est néanmoins fondamental d'avoir identifié **toutes** les clés au début du processus de normalisation.

3.4. Deuxième forme normale

2NF *Définition*

Une relation est en *2NF*∗ si elle est en *1NF*∗ et si tout attribut n'appartenant à aucune clé candidate ne dépend pas d'une partie seulement d'une clé candidate.

Echelle de salaire *Exemple*

Soit la relation Personne :

```
Personne(#NumeroEmployé, #Profession, Nom, Prénom, Salaire)
```

Soit les DF suivantes sur cette relation :

- NumeroEmployé, Profession→Nom
- NumeroEmployé, Profession→Prénom
- NumeroEmployé, Profession→Salaire
- Profession→Salaire

Personne n'est pas en 2NF car Profession (une partie de clé) détermine Salaire (un attribut qui n'appartient pas à une clé)

Pour avoir un schéma relationnel en 2NF, il faut alors décomposer Personne en deux relations :

```
Personne(#NumeroEmployé, #Profession=>Profession, Nom, Prenom)
Profession(#Profession, Salaire)
```

On remarque que ce schéma est en 2NF (puisque Salaire dépend maintenant fonctionnellement d'une clé et non plus d'une partie de clé).

On remarque aussi que la décomposition a préservé les DF, puisque nous avons à présent :

- Profession→Salaire (DF de la relation Profession)
- NumeroEmployé, Profession→Profession (par Réflexivité)
- NumeroEmployé, Profession→Salaire (par Transitivité)

Attention

La définition de la 2NF doit être vérifiée pour toutes les clés candidates et non seulement la clé primaire (dans le cas où il y a plusieurs clés).

Remarque

Si toutes les clés d'une relation ne contiennent qu'un unique attribut, et que la relation est en 1NF, alors la relation est en 2NF.

3.5. Troisième forme normale

3NF — **Définition**

Une relation est en *3NF*∗ si elle est en *2NF*∗ et si tout attribut n'appartenant à aucune clé candidate ne dépend directement que de clés candidates.

C'est à dire que toutes les *DFE*∗ vers des attributs n'appartenant pas à une clé, sont issues d'une clé.

Échelle de salaire et de prime — **Exemple**

Soit la relation Profession :

```
Profession(#Profession, Salaire, Prime)
```

Soit les DF suivantes sur cette relation :

- Profession→Salaire
- Profession→Prime
- Salaire→Prime

Cette relation n'est pas en 3NF car Salaire, qui n'est pas une clé, détermine Prime.

Pour avoir un schéma relationnel en 3NF, il faut décomposer Profession :

```
Profession(#Profession, Salaire=>Salaire)
Salaire(#Salaire, Prime)
```

Ce schéma est en 3NF, car Prime est maintenant déterminé par une clé.

On remarque que cette décomposition préserve les DF, car par transitivité, Profession détermine Salaire qui détermine Prime, et donc Profession détermine toujours Prime.

Clé candidate — **Attention**

La définition concerne toutes les clés candidates et non uniquement la clé primaire (*SQL avancé : Programmation et techniques avancées*∗, p.27).

Fondamental

Il est souhaitable que les relations logiques soient en 3NF. En effet, il existe toujours une décomposition sans perte d'information et préservant les DF d'un schéma en 3NF. Si les formes normales suivantes (*BCNF∗*, *4NF∗* et *5NF∗*) assurent un niveau de redondance encore plus faible, la décomposition permettant de les atteindre ne préserve plus systématiquement les DF.

Limite de la 3NF — *Remarque*

Une relation en 3NF permet des dépendances entre des attributs n'appartenant pas à une clé vers des parties de clé.

3NF et 2NF — *Complément*

Une relation en 3NF est forcément en 2NF car :

- Toutes les DFE vers des attributs n'appartenant pas à une clé sont issues d'une clé, ce qui implique qu'il n'existe pas de DFE, issues d'une partie de clé vers un attribut qui n'appartient pas à une clé.
- Il ne peut pas non plus exister de DFE issues d'une partie de clé vers un attribut appartenant à une clé, par définition de ce qu'une clé est un ensemble minimum.

On n'en conclut qu'il ne peut exister de DFE, donc a fortiori pas de *DF∗*, issues d'une partie d'une clé, et donc que toutes les DF issues d'une clé sont élémentaires.

3.6. Forme normale de Boyce-Codd

BCNF — *Définition*

Une relation est en *BCNF∗* si elle est en *3NF∗* et si les seules *DFE∗* existantes sont celles pour lesquelles une clé candidate détermine un attribut.

Employés — *Exemple*

Soit la relation Personne :

```
Personne(#N°SS, #Pays, Nom, Région)
```

Soit les DF suivantes sur cette relation :

- N°SS,Pays→Nom
- N°SS,Pays→Région
- Région→Pays

Il existe une DFE qui n'est pas issue d'une clé et qui détermine un attribut appartenant à une clé. Cette relation est en 3NF, mais pas en BCNF (car en BCNF toutes les DFE sont issues d'une clé).

Pour avoir un schéma relationnel en BCNF, il faut décomposer Personne :

```
Personne(#N°SS, #Region=>Region, Nom)
Region(#Region, Pays)
```

Remarquons que les DF n'ont pas été préservées par la décomposition puisque N°SS et Pays ne déterminent plus Région.

Simplicité — *Fondamental*

La BCNF est la forme normale la plus facile à appréhender intuitivement et formellement, puisque les seules DFE existantes sont de la forme K→A où K est une clé.

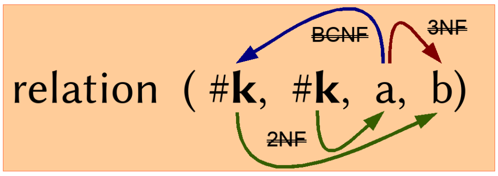
À retenir — *Méthode*

Pour prouver qu'une BD n'est pas redondante, on pourra se contenter de vérifier qu'elle est en BCNF.

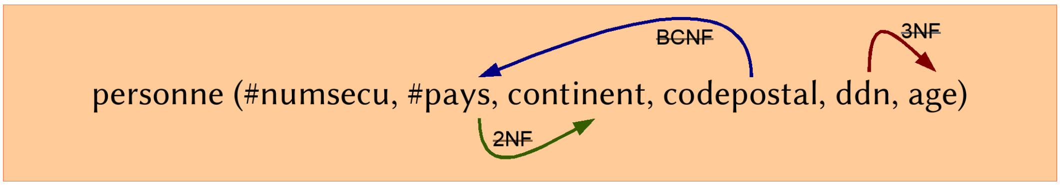
Non préservation des DF — *Attention*

Une décomposition en BCNF ne préserve pas toujours les DF.

3.7. Synthèse



Exemple



3.8. Exercice : A1, coulé !

Considérons le schéma de la relation suivante :

- `r (A, B, C, D, E)`

Cette relation est définie en extension par les tuples suivants :

A	B	C	D	E
a1	b2	c2	d3	e2
a1	b2	c2	d1	e4
a2	b3	c2	d1	e5
a2	b4	c5	d1	e5

Après avoir énoncé les DF et les clés, déterminer la forme normale du schéma ?

- ☐ 1NF
- ☐ 2NF
- ☐ 3NF
- ☐ BCNF

4. Bibliographie commentée sur la normalisation

Synthèses *Complément*

SQL2 SQL3, applications à Oracle ∗

On conseillera de lire le chapitre 2 (pages 42 à 49) qui propose une très bonne démonstration par l'exemple des problèmes posés par une mauvaise modélisation relationnelle.

Une description claire des formes normales, rendue simple et pratique grâce à des exemples représentatifs (chapitre 2).

Exercices

1. Exercice : De quoi dépend un cours ?

[15 min]

On considère le schéma relationnel R défini sur les attributs suivants : C : cours, P : professeur, H : heure, S : salle, E : étudiant, N : note.

Un nuplet (c, p, h, s, e, n) a pour signification que le cours c est fait par le professeur p à l'heure h dans la salle s pour l'étudiant e qui a reçu la note n.

L'ensemble E des dépendances fonctionnelles initiales est le suivant :

- C → P
- H,S → C
- H,P → S
- C,E → N
- H,E → S

Question 1

Donner la fermeture transitive F+ des dépendances fonctionnelles élémentaires engendrées par E.

Question 2

Quelle est la clé de la relation R ? Montrer qu'elle est unique.

2. Exercice : Cuisines et dépendances

[20 min]

On considère une relation R construite sur les attributs Propriétaire, Occupant, Adresse, Noapt, Nbpièces, Nbpersonnes, un nuplet (p, o, a, n, nb1, nb2) ayant la signification suivante : La personne o habite avec nb2 personnes l'appartement de numéro n ayant nb1 pièces dont le propriétaire est p.

Une analyse de cette relation nous fournit un ensemble initial E de dépendances fonctionnelles :

- occupant → adresse
- occupant → noapt
- occupant → nbpersonnes
- adresse, noapt → propriétaire
- adresse, noapt → occupant
- adresse, noapt → nbpièces

Question 1

Donner l'ensemble des DFE engendrées par E (fermeture transitive F+).

Question 2

Quelles sont les clés candidates de R ?

Question 3

Montrer que R est en 3NF de deux façons différentes (sans passer et en passant par la BCNF).

3. Test : Normalisation

Exercice 1

Soit la relation suivante et une couverture minimale des DF associée.

```
tUtilisateur (pklogin,mdp,nom,prenom,ville)
```

- pklogin→mdp,nom,prenom,ville
- nom,prenom→pklogin
- ville→nom

Sélectionner la ou les clés de cette relation.

- [ ] pklogin
- [ ] mdp
- [ ] nom
- [ ] prenom
- [ ] ville
- [ ] (pklogin, mdp)
- [ ] (pklogin, nom)
- [ ] (nom, prenom)
- [ ] (ville, nom)
- [ ] (nom, prenom, ville)

Exercice 2

Soit le schéma relationnel (on pose que les attributs A, B, C, D, X et Y sont atomiques) :

```
R1(A, B, C, D)
R2(X, Y)
```

Soit les dépendances fonctionnelles identifiées :

- A → C
- A,B → D
- X → Y

En quelles formes normales est ce schéma relationnel ?

- [ ] 1NF
- [ ] 2NF
- [ ] 3NF
- [ ] BCNF

Exercice 3

Soit le schéma relationnel suivant :

```
Personne (Nom, Prenom, Age, DateNaissance)
```

Soit les DF suivantes :

- Nom → Prenom
- Nom, Prenom → Age
- Prenom → DateNaissance, Nom
- DateNaissance, Age → Age, Nom
- Age, Nom → Age, DateNaissance
- DateNaissance → Age

Quelles sont les clés candidates pour la relation Personne ?

- [ ] Nom
- [ ] Prenom
- [ ] Age
- [ ] DateNaissance

Exercice 4

Soit la relation R (A:Int, B:Int, C:Int, D:Int, E:Int) et l'ensemble de *DFE*∗ suivant : {A → B ; A → C ; A → D ; A → E ; B → A ; B → C}

Sélectionner toutes les assertions vraies.

- [ ] A est une clé
- [ ] B est une clé
- [ ] C est une clé
- [ ] Le schéma est en 1NF
- [ ] Le schéma est en 2 NF
- [ ] Le schéma est en 3 NF
- [ ] Cet ensemble de DFE est une fermeture transitive.
- [ ] Cet ensemble de DFE est une couverture minimale.

Exercice 5

Soit le schéma relationnel (on pose que tous les attributs sont atomiques) :

```
Adresse (Numero, Rue, Ville=>Ville, Pays=>Ville)
Ville (Ville, Pays=>Pays)
Pays (Pays)
```

En quelles formes normales est ce schéma relationnel ?

- [ ] 1NF
- [ ] 2NF
- [ ] 3NF

- [ ] BCNF

Glossaire

Relation toute clé

En base de données, on appelle une relation toute clé une relation dont tous les attributs sont nécessaires pour constituer une clé.

Abréviations

1NF : First Normal Form
2NF : Second Normal Form
3NF : Third Normal Form
4NF : Fourth Normal Form
5NF : Fifth Normal Form
BCNF : Boyce-Codd Normal Form
BD : Base de Données
DF : Dépendance Fonctionnelle
DFE : Dépendance Fonctionnelle Elémentaire

Bibliographie

Celko Joe. *SQL avancé : Programmation et techniques avancées*. Vuibert, 2000.

Delmal Pierre. *SQL2 SQL3, applications à Oracle*. De Boeck Université, 2001.

Index