Démarche de conception de Bases de Données : Dépendances Fonctionnelles et Normalisation

HLIN511

Pascal Poncelet
LIRMM
Pascal.Poncelet@lirmm.fr
http://www.lirmm.fr/~poncelet

Introduction

- Il existe de nombreuses méthodes pour concevoir une base :
  - Empirique (DANGER)
  - Semi empirique (ex : entité-association)
  - Formelle (ex : normalisation relationnelle)

Introduction

- Objectif : trouver un bon schéma relationnel
- La qualité d'un schéma se mesure lors des opérations de mises à jour

AEROPORT(Num, Nom, Categorie, Salaire)

AEROPORT	Num	Nom	Categorie	Salaire
	1	DUPONT	PILOTE	40
	2	DURANT	MECANICIEN	15
	3	DUJARDIN	PILOTE	40
	4	DURATEAU	ACCUEIL	8

- hypothèse : la catégorie détermine le salaire

Introduction

- Anomalies de modification
  - Modification du salaire des pilotes : autant de modifications qu'il y a de pilotes
- Anomalies d'insertion
  - Pour stocker le salaire des contrôleurs il faut qu'il y ait au moins une catégorie d'employés dans cette catégorie (pas de clé primaire nulle)
- Anomalies de suppression
  - Si suppression de DURANT on perd l'information sur le salaire des mécaniciens

4

Introduction

- L'objectif est d'éliminer ces anomalies pour obtenir un bon schéma relationnel
- La solution : normaliser la relation en la décomposant en plusieurs relations
- Il faut répondre aux questions suivantes :
  - S'il y a redondance : comment décomposer la relation ?
  - Y a t'il de l'information perdue lors de la décomposition ?
- Pour y répondre : les dépendances fonctionnelles

5

Dépendances fonctionnelles

- Soit R(U), une relation avec U l'ensemble des attributs. Soit X, Y $\subset$ U, i.e. X et Y sont deux attributs ou ensembles d'attributs de R
- Il existe une dépendance fonctionnelle (DF) entre X et Y, notée $X \rightarrow Y$ si et seulement si

  $\forall$ t1,t2$\in$ R

  si t1(X) = t2(X) alors t1(Y) = t2(Y)

- Les dépendances fonctionnelles sont des propriétés sémantiques (du schéma de R), donc s'appliquent à toute extension de R

6

Dépendances fonctionnelles

- Dans notre exemple :
- AEROPORT (Num, Nom, Categorie, Salaire)
- Nous avons : Categorie ⟶ Salaire
- ∀ t1, t2 projetés sur Cagorie et Salaire nous avons :

Si t_1(Categorie) = t_2(Categorie) alors t_1(Salaire) = t_2(Salaire)

Categorie	Salaire
PILOTE	40
MECANICIEN	15
PILOTE	40
ACCUEIL	8

PILOTE ⟶ 40
MECANICIEN ⟶ 15
ACCUEIL ⟶ 8

7

Dépendances fonctionnelles

- Dans une relation tout attribut est en DF avec la clé primaire

Num ⟶ Nom (à tout Numéro correspond un Nom)
Num ⟶ Categorie (à tout Numéro correspond une Catégorie)
Num ⟶ Salaire (à tout Numéro correspond un Salaire)

8

Dépendances fonctionnelles

Num ⟶ Nom
Num ⟶ Categorie
Num ⟶ Salaire

- Vocabulaire :

Num détermine Nom, Categorie, Salaire
Nom est déterminé par Num
Categorie est déterminé par Num
Salaire est déterminé par Num

9

Exemple

- Soit l'extension suivante de R
- DF dans R ?

A	B	C
A_1	B_1	C_1
A_1	B_1	C_2
A_2	B_2	C_2
A_3	B_3	C_2
A_3	B_4	C_2

10

Exemple

- NON A → B car A_3 donne B_3 et B_4
- NON A → C car A_1 donne C_1 et C_2
- OUI B → A car même valeur de A_1 pour B_1
- NON B → C car B_1 donne C_1 et C_2
- NON C → A car A différents pour C_2
- NON C → B car B différents pour C_2

A	B	C
A_1	B_1	C_1
A_1	B_1	C_2
A_2	B_2	C_2
A_3	B_3	C_2
A_3	B_4	C_2

Remarque :
B_3 et B_4 donnent A_3 et A_3
Nous regardons la partie gauche
B → A et non pas A → B

11

Propriétés des DF

- Axiomes d'Armstrong
- P1 : Réflexivité
  - si Y ⊆ X alors X → Y (donc X → X) **DF triviale**

    Si A,B → A,B alors A,B → A et A,B → B
- P2 : Augmentation
  - Si X → Y alors X,Z → Y,Z où Z ⊆ U

    Si A,B → C alors A,B,C → C
    Si A,B → D alors A,B,C → CD
- P3 : Transitivité
  - Si X → Y et Y → Z alors X → Z

    Si A,B → C et C → D alors A,B → D

12

Propriétés des DF

- Règles d'inférences déduites des axiomes d'Armstrong
- P4 : Pseudo-transitivité
  - Si X → Y et Y, Z → W alors X,Z → W
- P5 : Union
  - Si X → Y et X → Z alors X → Y,Z
- P6 : Décomposition
  - Si X → Y,Z alors X → Y et X → Z

13

Exemple

- Soit R (A, B, C, D, E, G, H) et
  F= {A,B → C ; B → D ; C,D → E ; G → A ; D → H}
- Avec les axiomes d'Armstrong nous avons :

B → H
  - car B → D et D → H (P3)

B,G → C
  - car G → A et par augmentation (P2) on a B,G → A,B. De plus A,B → C donc par transitivité (P3) on a B,G → C

A,B → E
  - car B → D et par augmentation (P2) on a A,B → A,D donc par décomposition (P6) on a A,B → D or nous avons A,B → C par union (P5) on a A,B → CD et nous savons que C,D → E donc par transitivité (P3) on obtient A,B → E

14

Fermeture d'un ensemble de DF

- Soit F un ensemble de dépendances fonctionnelles sur R(U). Soit X → Y une DF.
- F implique X → Y, noté : F ⊨ X → Y, signifie que toute instance de relation sur R qui satisfait les dépendances dans F satisfait aussi X → Y
- Soit F ={B → D, D → H } sur R (A, B, C, D, E, G, H). Soit la dépendance fonctionnelle B → H.

F ⊨ B → H

15

Fermeture Transitive

- La fermeture transitive (ou clôture) d'un ensemble de dépendances fonctionnelles, F, est ce même ensemble enrichi (F^+) de toutes les dépendances fonctionnelles que l'on peut dériver en appliquant les axiomes d'Armstrong
- En d'autres termes : F^+ contient toutes les DF impliquées par F : $F^+ = \{X \longrightarrow Y \mid F \vDash X \longrightarrow Y\}$

16

Un exemple

- F = {A →B ; B →C}
- La fermeture transitive est en fait :

  F^+ = {A→B ; B→C ; A→C ; *A→A ; A→A,B ; A→A,C ; A→A,B,C ; A,B→A ; A,B→A,B ; A,B→A,C ; A,B→A,B,C ; A,B,C→A ; A,B,C→A,B ; A,B,C→A,C ; A,B→A,B,C ; … C→C ; B,C→C*}

- La fermeture contient trop de solutions triviales
  - Calcul des fermetures d'ensembles d'attributs

17

Un exemple

- La fermeture transitive d'un ensemble d'attributs X est notée X^+ elle contient tous les attributs qui dépendent des attributs dans X

$$[X]^+_F = \{A \mid F \mid = X \rightarrow A\}$$

$$F = \{A \rightarrow B \; ; \; B \rightarrow C\}, R(AC)$$

$$[A]^+_F = \{ A, B, C \}$$

A partit de A on peut déterminer l'ensemble des attributs : A →C est dans la fermeture de F

- A →C est dans la fermeture de F
- A est une surclé (et clé) de R

18

Calcul de la Fermeture Transitive d'un ensemble d'attributs

Entrée : F un ensemble de DF et X un ensemble d'attributs

Sortie : X^+ fermeture transitive de X

Algorithme:

1. Initialiser X^+ à X
2. Trouver une DF ($G \longrightarrow D$) $\in$ F possédant en partie gauche des attributs inclus dans X^+
3. Ajouter dans X^+ les attributs situés en partie droite de la DF
4. Répéter 2) et 3) jusqu'à ce que X^+ ne puisse plus évoluer

19

Exemple

- Soit F={$A \longrightarrow D$; $A,B \longrightarrow E$; $B,I \longrightarrow E$; $C,D \longrightarrow I$; $E \longrightarrow C$}. Calculer la fermeture sous F de AE.

- (étape 1) $AE^+ = \{A,E\}$
- (étape 2) Comme $A \longrightarrow D$ nous pouvons ajouter D à l'ensemble : $AE^+= \{A,D,E\}$
- (étape 3) comme $E \longrightarrow C$ nous pouvons ajouter C : $AE^+= \{A,C,D,E\}$
- (étape 4) répéter étape 2 et 3 comme C,D sont dans AE+ et que nous avons $C,D \longrightarrow I$ alors nous pouvons ajouter I: $AE^+ = \{A,C,D,E,I\}$
- Pas d'évolution possible. La fermeture transitive de AE est donc : $AE^+= \{A,C,D,E,I\}$

Initialisation de AE^+ avec AE

Trouver une DF ($G \longrightarrow D$) $\in$ F possédant en partie gauche des attributs inclus dans X^+. Ici A est inclus dans AE^+

Ajouter dans X^+ les attributs situés en partie droite de la DF. Ici C.

On peut répéter l'étape 2 et 3 car X^+ a été étendu. Ici I

Plus d'évolution possible. X^+ contient la fermeture transitive

20

Un autre Exemple

- F= {$A \longrightarrow C$; $A \longrightarrow D$; $B,C \longrightarrow A$; $E \longrightarrow B$; $E \longrightarrow D$}. Calculer la fermeture sous F de CE.

- (étape 1) $CE^+ = CE$
- (étape 2) Comme $E \longrightarrow B$ nous pouvons ajouter B à l'ensemble : $CE^+= BCE$
- (étape 3) Comme $E \longrightarrow D$ nous pouvons ajouter D : $CE^+= BCDE$
- (étape 4) on peut refaire étape 2 et 3. Comme {B,C} est inclus dans {B,C,D,E} et que $B,C \longrightarrow A$ nous avons $C,D \longrightarrow I$ alors nous pouvons ajouter I: $CE^+= ABCDE$
- Pas d'étape 2 ni d'étape 3 possible. La fermeture transitive de CE est donc : $CE^+= ABCDE$

21

Equivalence

- Deux ensembles de DF différents peuvent exprimer les même contraintes :

  F = { A,B → D ; D →C ; C →D ; A,B →C }

  G ={A,B →C ; D →C ; C →D}

- F et G sont équivalents (expriment les même contraintes). On note : F ≡ G
- G est plus « compacte »

22

Equivalence

- F est équivalent à G(F ≡ G) ssi $F^+ = G^+$

- Pour vérifier si F et G sont équivalents :
  1. Pour chaque df X→Z dans F :
     vérifier si X → Z est dans G+ (calculer X+ par rapport à G et vérifier si Z ⊆ X+)
  2. De façon similaire, pour chaque df X → Z dans G: vérifier si X → Z est dans F+

23

Dépendance fonctionnelle élémentaire

- Soit R(U) une relation, soit X et Y ⊂ U, tels que : X → Y. La dépendance fonctionnelle X → Y est dite élémentaire (ou totale) ssi :
  - Y n'est pas inclus dans X, i.e. Y = U-X (Y est le complémentaire de X dans U)
  - il n'existe pas X' ⊂ X tel que X → Y

- La seconde condition indique que X est « la plus petite quantité d'information donnant Y »
- Il n'y a pas d'attribut inutile dans la partie gauche

24

Dépendance fonctionnelle directe

- Soit R(U) une relation, soit X et Y $\subset$ U, tels que : X $\rightarrow$ Y.
- La dépendance X $\rightarrow$ Y est directe s'il n'existe pas Z dans R distinct de X et Y tel que X $\rightarrow$ Z et Z $\rightarrow$ Y
- la dépendance n'est pas obtenue par transitivité

25

Dépendance fonctionnelle triviale et simple

- Soit R(U) une relation, soit X et Y $\subset$ U, tels que : X $\rightarrow$ Y.
- La dépendance X $\rightarrow$ Y est triviale si Y – X est vide
- Une dépendance fonctionnelle est simple si elle ne comporte qu'un seul attribut en partie droite et si elle n'est pas triviale

  $X \rightarrow A_1,A_2,\ldots,A_n \Leftrightarrow \{X \rightarrow A_1 ; X \rightarrow A_2 ;\ldots;X \rightarrow A_n \}$
- Il est toujours possible de présenter les dépendances fonctionnelles sous forme simple (P6)

26

Couverture Minimale

- Couverture minimale d'un ensemble de DF : sous ensemble minimum de dépendances fonctionnelles élémentaires qui permettent de générer toutes les autres
- Tout ensemble de dépendances fonctionnelles possède une couverture minimale (pas forcement unique)

27

Algorithme pour Couverture Minimale

Entrée : F un ensemble de DF et X un ensemble d'attributs
Sortie : M : la couverture minimale de F
Algorithme :

1. Décomposer chaque DF pour avoir un seul attribut à droite (P6). (*les cibles de DF n'ont qu'un attribut - DF simples*)
2. Supprimer les attributs en surnombre à gauche :
Pour tout X → Y, s'il existe un Z⊆X tel que Z → Y alors remplacer X → Y par Z → Y (propriétés P1, P3 et P4) (*pas d'attribut inutile dans les DF de M – DF élémentaires*)
3. Supprimer les DF redondantes (qu'on peut obtenir par les axiomes d'Armstrong – DF directes)

28

Recherche DF élémentaires

- Soit F un ensemble de DF, soit f ∈ F, la DF A, B, C, D … → Y

  A est un **attribut inutile** dans f si on peut engendrer

  B, C, D, … → Y

  à partir des DF de F et des propriétés P1, P3 et P4 (réflexivité, transitivité, pseudo-transitivité)

29

Recherche DF élémentaires

- Exemple

F = {A → B ; A, B → C}

B est inutile dans A,B → C car A → C peut être généré

P4 : X → Y   Y,Z → W alors   X,Z → W
A → B   B,A → C alors   A,A → C

et donc A → C

30

Exemple

- Soit {A, B, C, D, E, F} un ensemble d'attributs
- L'ensemble des DF est composé de :
- F= {f1 : A, B → C, D ; f2 : C → D ; f3 : E → D ; f4 : F → E,D ; f5 : B → A ; f6 : E, F → F ; f7 : D → E}

31

Exemple

F= {f1 : A, B → C, D ; f2 : C → D ; f3 : E → D ; f4 : F → E,D ; f5 : B → A ; f6 : E, F → F ; f7 : D → E}

1) Eclatement des DF

(les cibles de df n'ont qu'un attribut)

f1 : g'1 : A, B → C et g''1 : A, B → D
f4 : g'4 : F → E et g''4 : F → D

2) Suppression des attributs inutiles (à examiner g'1, g''1 et f6)

(pas d'attribut inutile dans les DF de M)

Pour g'1 et g''1 A est étranger
P4 X → Y Y,Z → W alors X,Z → W
B → A A,B → C alors B → C
pour f6 E est étranger
F → E E,F → F F,F → F

32

Exemple

F= {f1 : A, B → C, D ; f2 : C → D ; f3 : E → D ; f4 : F → E,D ; f5 : B → A ; f6 : E, F → F ; f7 : D → E}
Etapes précédentes : g'1 : B → C ; g''1 : B → D ; f6 : F → F ; g'4 : F → E ; g''4 : F → D

3) Suppression des DF redondantes

(pas de DF redondantes dans M)

g''1 redondante car g'1 et f2
g''4 redondante avec g '4 et f3
f6 redondante par réflexivité

Couverture minimale :
M={g'1 : B → C ; f2 : C → D ; f3 : E → D ; g'4 : F → E ; f5 : B → A ; f7 : D → E}

33

Dépendances fonctionnelles et clés

- Une clé d'une relation $R(A_1, ..., A_n)$ est un sous ensemble X des attributs de la relation R tel que les deux conditions ci-dessous sont vérifiées :
  1. $X \rightarrow A_1, ..., A_n$
  2. Il n'existe pas de $Y \subset X$ tel que $Y \rightarrow A_1, ..., A_n$
- Un attribut clé et un attribut qui appartient à cette clé et un attribut non clé est un attribut qui n'y appartient pas
- Super clé : tout ensemble d'attributs satisfaisant la 1ère propriété constitue une super clé de R
  - Une super clé de R contient donc une clé de R
  - Une clé de R est une super clé minimale de R
  - Si X=U, la relation est dite « toute clé » : la clé est composée de l'ensemble des attributs

34

Exemple

A	B	C	D
A1	B1	C1	D1
A1	B2	C1	D2
A2	B2	C2	D3
A3	B1	C1	D2
A4	B4	C3	D2

- L'un des attributs peut-il jouer le rôle de clé ?
- Quelles associations d'attributs peuvent jouer ce rôle ?

35

Exemple

A	B	C	D
A1	B1	C1	D1
A1	B2	C1	D2
A2	B2	C2	D3
A3	B1	C1	D2
A4	B4	C3	D2

- A ne peut pas car A_1 détermine plusieurs B (B_1, B_2)
- B ne peut pas car B_1 détermine plusieurs A (A_1, A_3)
- C ne peut pas car C_1 détermine plusieurs B (B_1, B_2)
- D ne peut pas car D_2 détermine plusieurs C (C_1, C_3)
- A,B oui car pas deux fois la même occurrence
- A,C non car A_1, C_1 détermine plusieurs B (B_1, B_2)
- A,D oui car pas deux fois la même occurrence
- B,C non car B_1, C_1 détermine plusieurs A (A_1, A_3)
- B, D oui car pas deux fois la même occurrence
- C,D non car C_1, D_2 détermine plusieurs A (A_1, A_3)
- A,B,C oui car pas deux fois la même occurrence
- B,C, D oui car pas deux fois la même occurrence
- A,B,C,D oui car pas deux fois la même occurrence - ABCD est une super clé

36

Exemple Intuitif

- Soit la relation R(A,B,C,D,E) et F= {A→C, A→D, BC→A, E→B, E→D}. Calculez la liste des clés candidates de R, F^+

E doit faire partie de la clé parce qu'il n'apparaît jamais dans une partie des DF. Comme A, B, C et D interviennent à droite des DF, il est inutile de calculer A^+, B^+, C^+, D^+. Une clé candidate doit avoir comme fermeture {A,B,C,D,E}.

Calcul de la fermeture de E^+ = {B,D,E}. E ne peut donc pas être seul clé. Il faut ajouter les combinaisons des attributs avec E : Calcul des fermetures de AE^+, BE^+, CE^+, DE^+

Ne vérifient pas la fermeture : BE^+ = {B,D,E} ; DE^+ = {B,D,E}

Vérifient la fermeture : AE^+ = {A,B,C,D,E} ; CE^+ = {A,B,C,D,E} (voir exemple précédent)

Les clés candidates sont : AE^+ et CE^+

37

Recherche des clés

Entrée : ensemble des attributs et ensemble des DF

Sortie : ensemble de relations avec leurs clés

Algorithme :

1. Recherche de la couverture minimale M
2. Regroupement des DF de M
   - Réunir dans un même ensemble E_i toutes les DF ayant même source (autant de E_i que de source de DF différentes)
3. Regroupement des E_i
   - On regroupe dans un même ensemble les DF de E_i et E_j s'ils contiennent des DF réciproques (X → Y et Y → X)
4. Création des relations
   - Réunir dans un même ensemble E_i toutes les DF ayant même source (autant de E_i que de source de DF différentes)

Exemple

- Soit {A, B, C, D, E, F} un ensemble d'attributs
- L'ensemble des DF est composé de :
- F= {f1 : A,B → C,D ; f2 : C → D ; f3 : E → D ;
  f4 : F → E,D ; f5 : B → A ; f6 : E,F → F ;
  f7 : D → E}

Etape 1 (voir transparent précédent) :

M={g'1 : B → C ; f2 : C → D ;
f3 : E → D ; g'4 : F → E ;
f5 : B → A ; f7 : D → E}

39

Exemple

- Etape 2

  $E_1 = \{B \rightarrow C ; B \rightarrow A\}$ $E_2 = \{C \rightarrow D\}$

  $E_3 = \{E \rightarrow D\}$ $E_4 = \{F \rightarrow E\}$ $E_5 = \{D \rightarrow E\}$
- Etape 3

  E_1 ; E_2; $E'_3 = E_3 \cup E_5$; E_4
- Etape 4

  R_1 (B,*C*,A) ; R_2 (C, D) ; R_3 (E, D) D clé candidate ;
  R_4 (F, *E*)

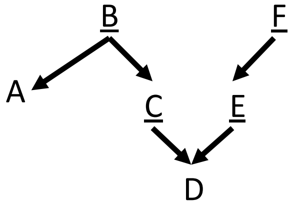
Graphes des dépendances fonctionnelles

- A partir du schéma de la base de données, il est possible de dessiner le graphe des dépendances fonctionnelles
- Le principe est le suivant :
  - Le ou les attributs clés primaires sont soulignés
  - Il y a une flèche du ou des attributs clés primaires vers les attributs non clés primaires

41

Exemple

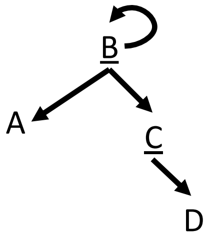
- R_1 (B,*C*,A) ; R_2 (C, D) ; R_3 (E, D) ; R_4 (F, *E*)



42

Exemple

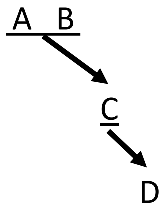
- Autojointure R_1 (B,*C*,A, *refB*) ; R_2 (C, D)



Exemple

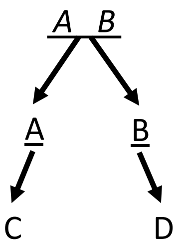
- Clé primaire multi-attributs

R_1 (A,B,*C*) ; R_2 (C, D)



Exemple

- Clé primaire multi-attributs avec relations complexes R_1 (*A*,*B*) ; R_2 (A, C) ; R_3 (B, D)



Le principe de la Normalisation

- Le point de départ est la relation universelle : l'ensemble de tous les attributs
- Objectif : une représentation canonique des données présentant un minimum de redondances à l'intérieur de chaque relation et un maximum d'indépendance entre les différentes relations
- Principe de la normalisation : remplacer une relation par d'autres relations afin que la jointure de ces relations permette de retrouver la relation initiale

46

Décomposition

- Critères attendus de la décomposition :
  - Décomposition sans perte d'information
  - Décomposition préservant les DF

47

Décomposition sans perte d'information

- Théorème de décomposition de Casey-Delobel (1973) :
- Soit R(X,Y,Z) une relation où X,Y,Z sont des ensembles d'attributs. Soit X ⟶ Y une DF vérifiée dans R.
  - Alors il existe R1 et R2 deux relations telle que : R1(X, Y) et R2 (X,Z) et R=JOINTURE(R1,R2/R1.X=R2.X)
- La décomposition de R dans les deux relations R1 et R2 est garantie sans perte d'information

48

Décomposition sans perte d'information

- Théorème de Heath :

Toute relation R(X,Y,Z) est décomposable sans perte d'information en R1(X,Y) et R2(X,Z) s'il existe une DF telle que X → Y

49

Décomposition préservant les DF

- La décomposition de R (A, F) en R1(A1, F1), R2(A2, F2) est une décomposition qui préserve les dépendances fonctionnelles ssi $F^+=(F1 \cup F2)^+$
- Soit la relation Entreprise (Ville, Rue, Code) et

F = Ville, Rue → Code ; Code -> Ville

Ville	Rue	Code
MONTPELLIER	COMEDIE	34000
MONTPELLIER	GARE	34000

50

Décomposition préservant les DF

- La décomposition de Entreprise en R1(Ville, Code) et R2(Rue, Code) évite la redondance Ville, Code et est une décomposition qui est sans perte d'information mais qui elle ne préserve pas la dépendance fonctionnelle :

Ville, Rue → Code

R1

Ville	Code
MONTPELLIER	34000

R2

Rue	Code
COMEDIE	34000
GARE	34000

51

Première Forme Normale

- Une relation est en 1FN ssi tous ses attributs sont atomiques (mono-valués)

PERSONNE	Num	Nom	Prénoms
	1	DUPONT	Jean, Paul, Jacques
	2	DURANT	Pierre, Patrick, Eric
	3	DUJARDIN	Marie, Emilie

LIVRE	Code	Titre	Auteur
	3A	Tintin	Hergé
	3B	Astérix	Goscigny, Uderzo

52

Première Forme Normale

- Comment normaliser en 1FN ? 2 solutions
- Créer autant d'attributs que le nombre maximum de valeurs (stockage horizontal)
  Personne (Num, Nom, Prenom1, Prenom2, Prenom3)
- Créer une nouvelle relation comprenant la CP de la relation initiale et l'attribut multi-valué
- Attention : éliminer l'attribut de la relation initiale
  Livre (Code, Titre) - Auteur (*Code*, NomAuteur)
- Avantage vs inconvénients ?

53

Deuxième Forme Normale

- Une relation est en seconde forme normale (2FN) ssi
  - Elle est en 1FN
  - Tout attribut n'appartenant pas à la clé primaire est en DF totale avec la clé

EMPRUNT (NumAbonné, NumLivre, Nom, DateEmprunt)

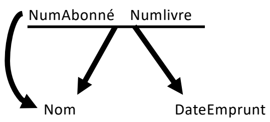
AUTEUR (NumAuteur, Nom, Adresse)

- EMPRUNT et AUTEUR en 2FN ?

54

Deuxième Forme Normale

EMPRUNT (NumAbonné, NumLivre, Nom, DateEmprunt)
avec NumAbonné ⟶ Nom



55

Deuxième Forme Normale

EMPRUNT (NumAbonné, NumLivre, Nom, DateEmprunt)
avec NumAbonné ⟶ Nom

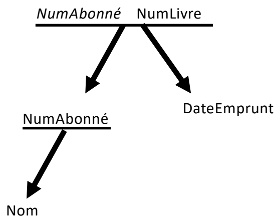
- Comment normaliser ?
- Isoler la DF responsable dans une nouvelle relation. Elle devient CP dans la relation initiale
- Eliminer l'attribut cible de la DF dans la relation initiale

EMPRUNT (*NumAbonné*, NumLivre, DateEmprunt)
ABONNE (NumAbonné, Nom)

56

Deuxième Forme Normale

EMPRUNT (*NumAbonné*, NumLivre,DateEmprunt)
ABONNE (NumAbonné, Nom)



57

Deuxième Forme Normale

- Attention : la seconde forme normale implique que :

« Tout attribut n'appartenant pas à la clé primaire est en DF totale avec la clé »

- Ceci doit être vrai pour les clés candidates
- A l'origine C. Date avait précisé que pour des raisons de simplicité, il suppose que chaque relation a une seule clé candidate …

58

Troisième Forme Normale

- Une relation est en 3FN ssi
- Elle est en 2FN
- Elle ne contient pas de DF transitive entre attributs non clés

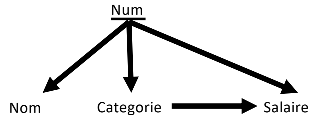
EMPRUNT (<u>*NumAbonné*, NumLivre</u>, DateEmprunt)
ABONNE (<u>NumAbonné</u>, Nom)
AEROPORT(<u>Num</u>, Nom, Categorie, Salaire)

- EMPRUNT, ABONNE, AEROPORT en 3NF ?

59

Troisième Forme Normale

AEROPORT(<u>Num</u>, Nom, Categorie, Salaire)
avec Categorie ⟶ Salaire



60

Troisième Forme Normale

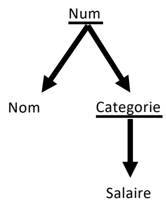
AEROPORT(Num, Nom, Categorie, Salaire)
avec Categorie ⟶ Salaire

- Comment normaliser en 3FN ?
- Isoler la DF transitive dans une nouvelle relation
- Eliminer l'attribut cible de la DF dans la nouvelle relation

PILOTE (Num, Nom, *Categorie*)
GRILLE (Categorie, Salaire)

Troisième Forme Normale

PILOTE (Num, Nom, *Categorie*)
GRILLE (Categorie, Salaire)



Résultat de la normalisation en 3FN

- Théorème :

toute relation R admet au moins une décomposition en 3FN telle que:

  - La décomposition préserve les DF
  - Toutes les composantes sont en 3FN

- Conséquences : Il est souhaitable que les relations soient en 3FN car il existe toujours une décomposition sans perte d'information et préservant les DF d'un schéma en 3FN

Exemple

- Soit R(A,B,C) avec les DF {A → B ; B → C} et A,B,C monovalués
- clé primaire de R ?
- forme normale de R ?
- l'extension suivante est-elle possible pour R ?
- Proposer une décomposition en 3FN pour R

A	B	C
A1	B1	C1
A2	B2	C2
A3	B2	C1
A4	B3	C3

64

Exemple

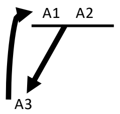
- Soit R(A,B,C) avec les DF {A → B ; B → C} et A,B,C monovalués
- clé primaire de R ? **A**
- forme normale de R ? **2FN car B → C**
- l'extension suivante est-elle possible pour R ? **NON car B2 donne C2 et C1**
- Proposer une décomposition en 3FN pour R :
  R1 (A,B) et R2 (B,C)

A	B	C
A1	B1	C1
A2	B2	C2
A3	B2	C1
A4	B3	C3

65

La forme Boyce and Codd

- Une relation est en BCFN (Boyce and Codd Normal Form) ssi elle est en 3FN et qu'aucun attribut de la clé ne dépend d'un attribut non clé. R (A1, A2, A3) est en BCFN s'il n'existe pas A3 → A1



66

La forme Boyce and Codd

- Théorème :

Toute relation admet une décomposition en BCFN sans perte d'information

Une décomposition BCFN ne conserve pas les DF

67

La forme Boyce and Codd

COURS (Matiere, Classe, Professeur) avec :

- Un professeur n'enseigne qu'une seule matière
- Une classe n'a qu'un seul enseignant par matière

Matiere, Classe → Professeur et

Professeur → Matiere

- La relation est en 3NF mais il existe une DF entre Professeur (non clé) et Matiere (partie de la clé)

68

La forme Boyce and Codd

Décomposition :

- SPECIALITE (Professeur, Matiere)
- ENSEIGNANT (Classe, Professeur)

- Décomposition sans perte d'information mais perte de la DF

Matiere, Classe → Professeur

- La DF ne peut pas être prise en compte par le SGBD. Nécessité d'avoir un trigger ou un programme à côté

Rappel

AEROPORT(Num, Nom, Categorie, Salaire)

AEROPORT	Num	Nom	Categorie	Salaire
	1	DUPONT	PILOTE	40
	2	DURANT	MECANICIEN	15
	3	DUJARDIN	PILOTE	40
	4	DURATEAU	ACCUEIL	8

- hypothèse : la catégorie détermine le salaire

Souvent au niveau BCFN et 3NF

- Plus d'anomalie de stockage
- Modification : modification du salaire des pilotes pour tous
- Insertion : on peut stocker le salaire d'un contrôleur sans avoir un employé de cette catégorie
- Suppression : si DURANT est supprimé on conserve l'information sur le salaire des mécaniciens

Dépendances multi-valuées

- Soit R (X, Y, Z) une relation. On dit que X↠Y (X multi détermine Y ou il y a une dépendance multi-valuée de Y sur X) si pour toute extension de R (X↠Y) :

A chaque valeur de X correspond toujours le même ensemble de valeurs de Y et cet ensemble de valeurs ne dépend pas de Z

Dépendances multi-valuées

- Un étudiant peut faire plusieurs sports et parler plusieurs langues

R (<u>NumEtudiant, Sport, Langue)</u>

NumEtudiant	Sport	Langue
1	FOOTBALL	FRANCAIS
1	TENNIS	ANGLAIS
1	NATATION	FRANCAIS
1	TENNIS	FRANCAIS

- NumEtudiant ↠ Sport, Langue
- Répétition de l'information : 1 – FRANCAIS, 1 - TENNIS

73

Dépendances multi-valuées

- Les dépendances multi-valuées sont une généralisation des dépendances fonctionnelles :

si X ⟶ Y alors X↠Y

- De même que pour les DF, une dépendance multi-valuée D est déductible de F si elle est obtenue par application des axiomes d'Armstrong

74

D'autres formes normales

- Il existe d'autres formes normales mais elles ne sont généralement peu utilisées car elles se font au dépend souvent de la perte de DF et en outre elles ont tendance à éclater complétement les relations
- Coût excessif pour les opérations de jointures

75

Test de validité de la décomposition

- Vérification qu'une décomposition est sans perte d'information
- Soit $R(A_1, A_2, \ldots, A_n)$ une relation à n attributs décomposée, par normalisation en $R_1, R_2, \ldots R_m$

- 1ère étape : création du tableau
  - Création d'un tableau dont les lignes correspondent aux relations $R_1, \ldots, R_m$ et les colonnes les n attributs de la relation initiale
  - A l 'intersection d'une ligne i et d'une colonne j, mettre α_j si $Aj \in Ri$ et $\beta_{i,j}$ sinon

76

Test de validité de la décomposition

- 2nd étape : unification
  - On considère le tableau comme l'extension de R. On examine les DF sur ce tableau
  - Si une DF n'est pas vérifiée, on unifie les valeurs de la cible aussi : si l'une des valeurs est α_j et les autres des $\beta_{i,j}$, on remplace les $\beta_{i,j}$ par des α_j

77

Test de validité de la décomposition

- 3ième étape : validation
  - Si le tableau contient au moins 1 ligne ne comportant que des α, la décomposition est sans perte d'information sinon elle est avec perte

78

Exemple

R (NumEt, NomEt, CodeUV, NoteTest, NoteCC)
avec

NumEt → NomEt

NumEt, CodeUV → NoteTest, NoteCC

- Décomposé en

R1 (NumEt, NomEt)

R2 (NumEt, CodeUV, NoteTest, NoteCC)

79

Exemple

R (NumEt, NomEt, CodeUV, NoteTest, NoteCC)
R1 (NumEt, NomEt)
R2 (NumEt, CodeUV, NoteTest, NoteCC)

Création du tableau

R	NumEt	NomEt	CodeUv	NoteTest	NoteCC
R1					
R2					

Projection des relations R1 et R2. α_i indique que l'attribut est dans la relation à la ieme colonne

R	NumEt	NomEt	CodeUv	NoteTest	NoteCC
R1	α_1	α_2	$\beta_{1,3}$	$\beta_{1,4}$	$\beta_{1,5}$
R2	α_1	$\beta_{2,2}$	α_3	α_4	α_5

80

Exemple

R (NumEt, NomEt, CodeUV, NoteTest, NoteCC)
R1 (NumEt, NomEt)
R2 (NumEt, CodeUV, NoteTest, NoteCC)

Vérification de la DF : NumEt → NomEt. Si α_i un donne un $\beta_{l,m}$ unifier : mettre un α_m

R	NumEt	NomEt	CodeUv	NoteTest	NoteCC
R1	α_1	α_2	$\beta_{1,3}$	$\beta_{1,4}$	$\beta_{1,5}$
R2	α_1	$\cancel{\beta_{2,2}}$ α_2	α_3	α_4	α_5

Il existe une ligne qu'avec des α alors la décomposition est sans perte d'information

81

Exemple

R (NumEt, NomEt, CodeUV, NoteTest, NoteCC) avec

NumEt → NomEt

NumEt, CodeUV → NoteTest, NoteCC

Décomposé en :

R1 (NumEt, NomEt, NoteTest)

R2 (NumEt, CodeUV, NoteCC)

Avec perte ou sans perte ?

82

- Des questions ?

83