

Normalisation des relations

PARTIE I

Dépendances et formes normales

Patricia Serrano Alvarado
A partir des slides de Sylvie Cazalens

Dpt Informatique
Université de Nantes

TODO avant la séance

- ◆ Lire ce polycopié
- ◆ Visionner
 - ◆ Vidéo Stanford - DB design <https://www.youtube.com/watch?v=DFnAakJ4FDg>
 - ◆ Vidéo Stanford - Functional dependencies <https://www.youtube.com/watch?v=Mkm1h5AtsXI>
 - ◆ Vidéo Stanford - BCNF <https://www.youtube.com/watch?v=mfVCesoMaGA>
 - ◆ Vidéo U. Houston - CV(DF) - partie 1 <https://www.youtube.com/watch?v=CL3jTz6UPbY>
 - ◆ Vidéo U. Houston - CV(DF) - partie 2 <https://www.youtube.com/watch?v=IiEAfRoHIH8>
 - ◆ Vidéo U. Houston - Keys https://www.youtube.com/watch?v=s1DNVWKeQ_w

Formes normales : pourquoi ?

- ◆ Lors de la gestion de la base, certaines relations peuvent poser un certain nombre de problèmes.
- ◆ Les formes normales caractérisent les relations qui présentent moins de problèmes que d'autres.
- ◆ Les formes normales sont définies à partir de l'analyse des « liens » entre les valeurs des attributs de la relation : les dépendances.

Problèmes (anomalies)

- ◆ **de redondance** : la même valeur d'un sous-ensemble d'attributs apparaît plusieurs fois.
- ◆ **à l'ajout** : l'ajout de certaines informations n'est possible que si d'autres sont présentes.
- ◆ **à la suppression** : la suppression de certaines informations, entraîne celle d'autres infos (que l'on aurait bien aimé conserver !)
- ◆ **à la mise à jour** : la modification d'une info doit être répercutée autant de fois qu'elle apparaît dans la relation.

Exemple (B. Defude, C. Date)

Produit	Quantité	Couleur	Fournisseur	Adresse
parapluie	110	rouge	Labaleine	Paris
chapeau	50	vert	Lemelon	Nantes
ceinture	65	noir	ToutCuir	Nantes
parasol	15	jaune	Labaleine	Paris
ombrelle	5	rouge	Labaleine	Paris
ceinture	25	vert	Letour	Lyon

Exemple

- ◆ **Redondance** : L'information « le fournisseur Labaleine est sur Paris » apparaît autant de fois que l'on a commandé de produit à ce fournisseur.
- ◆ **Mise à jour** : Si « Labaleine » change d'adresse, il faudra modifier autant de tuples qu'il y a de commandes.
- ◆ **Suppression** : Si l'on supprime le produit « ceinture » on supprime toutes les infos relatives au fournisseur « Letour ».
- ◆ **Ajout** : Pour insérer un nouveau fournisseur, il faut lui avoir commandé un produit.

Liens entre les attributs ?

- ◆ Entre «fournisseur» et «adresse» ?
 - ◆ Si on connaît le fournisseur, on connaît l'adresse correspondante.
 - ◆ Dépendance fonctionnelle entre fournisseur et adresse mais pas dans le sens adresse vers fournisseur.
- ◆ Entre « produit » et « fournisseur » ?
 - ◆ Si on connaît le produit, on ne connaît pas forcément le fournisseur.
 - ◆ Pas de dépendance fonctionnelle entre produit et fournisseur car ceinture fournit par 2 fournisseurs différents

Notations et conventions

- ◆ U : ensemble d'attributs ;
- ◆ $R(U)$: schéma de relation ;
- ◆ Par abus de langage, $R(U)$ parfois appelé « relation » ou R .
- ◆ r : un ensemble de tuples particulier (instance du schéma $R(U)$)
- ◆ X, Y, Z, W : sous-ensemble d'attributs de U , non vides.
- ◆ A, B, C : attributs particuliers.

Dépendance fonctionnelle (df)

- ◆ Soit $R(U)$ un schéma de relation ;
- ◆ soient X et Y deux sous-ensembles de U .
- ◆ La dépendance fonctionnelle
- ◆ $X \rightarrow Y$ est vraie sur R ssi quelle que soit r instance de R , $X \rightarrow Y$ est vrai dans r.
- ◆ i.e. ssi tous les tuples de R qui ont même valeur pour X ont même valeur pour Y

CONT.

- ◆ Si $X \rightarrow Y$ on dit
 - ◆ X détermine Y ou
 - ◆ Y dépend fonctionnellement de X .
- ◆ Une df est une propriété qui doit être extraite de la connaissance que l'on a de l'application.
- ◆ L'ensemble des dépendances fonctionnelles **DF** vérifiées constitue des informations à rajouter au schéma pour affiner la caractérisation.
- ◆ $\langle R(U), DF \rangle$

Dépendance élémentaire (dfe)

- Soit $\langle R(U), DF \rangle$ et soit la dépendance fonctionnelle $X \rightarrow A$ appartenant à DF , avec A non inclus dans X (i.e., pas trivial).
- $X \rightarrow A$ est élémentaire, s'il n'existe pas de sous-ensemble de X qui détermine A .

On dit que A dépend pleinement de X

Exemples

- $A,B \rightarrow B$ est une df trivial et donc pas dfe
B ne dépend pas pleinement de A,B.
- Si $A,B \rightarrow C$ et $B \rightarrow C$ alors $A,B \rightarrow C$ pas dfe
C ne dépend pas pleinement de A,B.

Exercice

Trouver les dfs de cette instance

Notes									
noetu	nom	prenom	diplôme	codemat	titre	resp	cc	exam	
27845E	Dupont	Isabelle	L3 info	X6I0050	BD2	E512	15	12	
27845E	Dupont	Isabelle	L3 info	X6I0010	Prog fonct	E860	10	11	
27845E	Dupont	Isabelle	L3 info	X6I0030	RO	E100	11	13,5	
34561C	Legarec	Marc	L3 info	X6I0050	BD2	E512	10,2	11	
34561C	Legarec	Marc	L3 info	X6I0010	Prog fonct	E860	10	13	
34561C	Legarec	Marc	L3 info	X6I0030	RO	E100	12	10	
45678D	Martin	Robert	L3 info	X6I0050	BD2	E512	10	8	
45678D	Martin	Robert	L3 info	X6I0030	RO	E100	5	5	
64289C	Dupont	Letitia	L3 miage	X6IM020	IHM	E350	15	10	
64289C	Dupont	Letitia	L3 miage	X6IM010	Gestion	E670	12,5	10	
64289C	Dupont	Letitia	L3 miage	X6I0050	BD2	E512	13,1	12	
23456E	Robert	Christine	L3 miage	X6IM010	Gestion	E670	7	2	
23456E	Robert	Christine	L3 miage	X6I0080	Fichier	E900	14	13,8	
98076E	Robert	Christine	L3 miage	X6IM020	IHM	E350	12	12,8	

Règles de dérivation d'Armstrong

- ◆ Réflexivité
 - ◆ Si Y est inclus dans X alors $X \rightarrow Y$
- ◆ Transitivité
 - ◆ Si $X \rightarrow Y$ et $Y \rightarrow Z$ alors $X \rightarrow Z$
- ◆ Augmentation
 - ◆ Si $X \rightarrow Y$ alors $X, Z \rightarrow Y, Z$

Règles de dérivation déduites

- ◆ Split
 - ◆ Si $A \rightarrow BC$ alors $A \rightarrow B$ et $A \rightarrow C$
- ◆ Union
 - ◆ Si $A \rightarrow B$ et $A \rightarrow C$ alors $A \rightarrow BC$
- ◆ Trivial
 - ◆ Si $X \rightarrow Y$ et Y est sous-ens de X alors $X \rightarrow X \cup Y$

cont.

- ◆ Pseudo-transitivité
 - ◆ Si $X \rightarrow Y$ et $W, Y \rightarrow Z$ alors $W, X \rightarrow Z$
- ◆ Décomposition
 - ◆ Si $X \rightarrow Y$ et Z est sous-ensemble de Y alors $X \rightarrow Z$

Fermeture d'un ensemble d'attributs X^+

- Etant donné un ensemble d'attributs X , quels sont tous les autres attributs qui sont déterminés par X ?
- Soit $\langle R(U), DF \rangle$; soit X inclus dans U ,
- X^+ est l'ensemble des attributs A tel que $X \rightarrow A$ est déductible de DF .
(équiv. à $X \rightarrow A$ appartient à DF^+)

Algorithme de calcul de X^+

1. Début $X^+ \leftarrow X$
2. Répéter
3. Pour chaque df $Y \rightarrow Z$ de DF faire
4. Si Y inclus dans X^+ , alors $X^+ \leftarrow X^+ \cup Z$
5. Jusqu'à ce que X^+ ne soit plus modifiable

Clés

- ◆ Ensemble minimal d'attributs qui détermine tous les autres.
- ◆ X est une clé de $\langle R(U), DF \rangle$ ssi :
 1. $X \rightarrow U$ est déductible de DF ($X^+ = U$)
 2. Il n'existe pas Y , Y strictement inclus dans X tel que $Y \rightarrow U$
- ◆ Il peut y avoir plusieurs clés. Un attribut qui appartient à une clé est parfois appelé « attribut clé ».

Calcul des clés

1. A l'aide d'un graphe de df visualiser les attributs X qui ne sont pas déterminés par aucun autre attribut
2. Si selon DF, X satisfait 1 et 2 alors X est une clé
3. Sinon, pour Y sous-ensemble de U, tester $(X,Y)^+$ jusqu'à satisfaire 1 et 2
4. Chercher toutes les clés possibles

Fermeture d'un ensemble de df (DF^+)

- Soit DF un ensemble de df.
- La fermeture (ou clôture) DF^+ de DF est l'ensemble de toutes les df déductibles à partir de DF avec les règles de dérivation.
- La fermeture comprend toutes les dfs déductibles, y compris les plus triviales.

Couverture minimale de DF

Sous ensemble minimal de DF permettant de générer toutes les autres.

Soit DF un ensemble de df.

Une couverture minimale de DF, noté $CV(DF)$ est un ensemble de df tel que :

- $CV(DF)^+ = DF^+$
- Toutes les df de $CV(DF)$ sont élémentaires
- Quel que soit $X \rightarrow A$ de $CV(DF)$, $(CV(DF) - \{X \rightarrow A\})^+ \neq DF^+$

Calcul de CV(DF)

1. Toutes les dépendances doivent être élémentaires ; les décomposer si nécessaire
2. Eliminer les attributs superflus du côté gauche de la df
3. Eliminer les dfs redondantes

Comment faire le pas 2

- ◆ Prendre une df avec coté gauche > 1 attributs (supposons 2 ici)
- ◆ Choisir un attribut côté gauche et calculer sa fermeture :
 - ◆ on peut éliminer l'autre attribut si
 - ◆ l'attribut de droite de la df apparaît dans le résultat ou
 - ◆ il apparaît dans le résultat de la fermeture.

Comment faire le pas 3

- ◆ On peut éliminer une df si par transitivité elle est préservée (donc elle n'est pas nécessaire)
- ◆ Ensuite, analyser DF, une df à la fois
 - ◆ Calculer la fermeture du côté gauche sans prendre en compte la df en cours d'analyse
 - ◆ Si dans le résultat apparaît l'attribut de droite alors la df est éliminée (car elle est préservée même si la df en analyse n'existe pas)

Les formes
normales

Formes normales

1 et 2

- ◆ 1FN : tous les attributs sont atomiques
- ◆ Il ne peut y avoir de listes de valeurs
- ◆ 2FN : 1FN + tous les attributs non clés ne dépendent pas d'une partie d'une clé
 - ⇒ Les attributs non clé dépendent pleinement d'une clé entière.

2FN

- ◆ La 2FN permet d'assurer l'élimination de certaines redondances en garantissant qu'aucun attribut n'est déterminé seulement par une partie de la clé.
- ◆ ~~PC \rightarrow NC~~

3e forme normale (v2)

- 2FN + tout attribut non clé ne dépend pas d'un autre attribut non clé
 - ⇒ les attributs non clés ne dépendent pas entre eux.
- ~~NC \rightarrow NC~~

3FN n'est pas suffisant

- ◆ Quid des dépendances entre parties de clés entre elles et entre attributs non clés et parties de clés.

Boyce-Codd-Kent (FNBCK ou BCNF)

- $\langle R(U), DF \rangle$ est en FNBCK si chaque fois que $X \rightarrow A$ (A n'appartenant pas à X) est vérifiée, X est une clé de R .
- ~~PC \rightarrow PC~~ et ~~NC \rightarrow PC~~
- Remarque (" $<$ " plus normalisé) :
 - FNBCK $<$ 3FN $<$ 2FN $<$ 1FN

Récap

- ◆ C→NC Le Graal
- ◆ 1FN considère que des valeurs atomiques
- ◆ 2FN élimine ~~PC→NC~~
- ◆ 3FN élimine ~~NC→NC~~
- ◆ FNBCK élimine ~~PC→PC~~ ; ~~NC→PC~~

C : attributs clé ; NC : attributs non-clé ; PC : attributs partie de clé

Quoi faire si une relation n'est pas normalisée ?

Découper la relation en plusieurs relations à l'aide des algorithmes de normalisation.

Normalisation des relations

PARTIE II
Algorithmes de Normalisation

Normalisation d'une relation

La normalisation dans une forme normale « nf » d'une relation R est le remplacement de la relation par plusieurs relations qui sont dans la forme normale « nf » et qui, « globalement » conservent **exactement** les informations contenues dans R.

Plan

- ◆ Décomposition d'une relation
- ◆ Décomposition sans perte de df (**spdf**)
- ◆ Décomposition sans perte d'information (**spi**)
- ◆ Algorithmes de normalisation
 - ◆ Par synthèse (algorithme de Bernstein)
 - ◆ Par décomposition

3FN et FNBCK

- ◆ Ces formes normales garantissent que les anomalies ne seront pas présentes
 - ◆ de redondance,
 - ◆ de mise à jour,
 - ◆ d'ajout et
 - ◆ de suppression
- ◆ Mais, peut-on
 - ◆ retrouver l'instance initiale ?
 - ◆ ne pas perdre de dépendances fonctionnelles ?

Restriction d'un ensemble de df ($DF|_X$)

- Soit le schéma de relation $\langle R(U), DF \rangle$ et soit X un ensemble d'attributs de U
- La restriction de DF à l'ensemble X , notée $DF|_X$ est constituée de l'ensemble des dfs de DF^+ formées d'attributs de X uniquement.

Décomposition

- Une **décomposition** de $\langle R(U), DF \rangle$ est le remplacement de $\langle R(U), DF \rangle$ par un ensemble de schémas
- $\langle R_1(U_1), DF|_{U_1} \rangle, \dots, \langle R_n(U_n), DF|_{U_n} \rangle$ où
 - pour tout i , U_i est sous-ensemble de U .
 - $R(U_i)$ obtenu par projection de $R(U)$ sur U_i .
 - L'union des U_i est égale à U .

Décomposition sans perte de dfs (spdf)

- ◆ Définition :

La décomposition de $\langle R(U), DF \rangle$ en

$\langle R_1(U_1), DF|_{U_1} \rangle, \dots, \langle R_n(U_n), DF|_{U_n} \rangle$

est sans perte de dépendances fonctionnelles (spdf) ssi

La fermeture de l'union des $DF|_{U_i}$ ($1 \leq i \leq n$) est égale à DF^+

Décomposition sans perte d'information (spi)

- ♦ Définition :

La décomposition de $\langle R(U), DF \rangle$ en

$\langle R_1(U_1), DF|_{U_1} \rangle, \dots, \langle R_n(U_n), DF|_{U_n} \rangle$

est sans perte d'information (spi) ssi

quelle que soit l'instance r de R , r est égale à la jointure naturelle des r_i ($1 \leq i \leq n$), chaque r_i étant obtenu par projection de r sur R_i .

Théorème pour spi

Soit $\langle R(U), DF \rangle$; $Z = U - \{X, Y\}$

- Si $X \rightarrow Y$ appartient à DF alors la décomposition de $\langle R(U), DF \rangle$ en $\langle R1(X, Y), DF|_{\{X, Y\}} \rangle$ et $\langle R2(X, Z), DF|_{\{X, Z\}} \rangle$ est spi.
- Inversement, si la décomp. de $\langle R(U), DF \rangle$ en $\langle R1(X, Y), DF|_{\{X, Y\}} \rangle$ et $\langle R2(X, Z), DF|_{\{X, Z\}} \rangle$ est spi alors $X \rightarrow Y$ ou $X \rightarrow Z$ appartient à DF^+

Liens spdf et spi

- ◆ Il n'y a aucun lien entre les propriétés « sans perte d'information » et « sans perte de dépendance fonctionnelle ».
- ◆ Une décomposition spi n'est pas forcément spdf
- ◆ et vice-versa.

Normalisation : résultats

Théorème 1

Toute relation en 1FN admet une décomposition en 3FN qui est à la fois spi et spdf.

Théorème 2

Toute relation en 1FN admet une décomposition en FNBCK qui est spi.

Algorithmes de normalisation

objectif

- Obtenir de manière automatique un ensemble de relations vérifiant la forme normale souhaitée pour remplacer une relation qui ne la vérifie pas.
- Deux types d'algorithmes :
 - Normalisation « **par synthèse** » ça donne 3FN
 - Normalisation « **par décomposition** » ça donne BCNF

Normalisation par synthèse

- On synthétise des schémas de relations à partir de l'ensemble des attributs et des dépendances fonctionnelles.
- L'algorithme de Bernstein est le plus utilisé.

Algorithme de Bernstein

- Entrée : $\langle R(U), DF \rangle$
- Sortie : une décomposition $\langle R_1(U_1), DF|U_1 \rangle, \dots, \langle R_n(U_n), DF|U_n \rangle$, à la fois spi et spdf où toutes les relations sont en 3FN.

cont.

1. Calculer $CV(DF)$, et les clés. Si R est en 3FN on s'arrête.
2. Partitioner $CV(DF)$ en groupes DF_i ($1 \leq i \leq k$) tels que toutes les df d'un même groupe aient même partie gauche.
3. Construire un schéma $\langle R_i(U_i), DF_i \rangle$ pour chaque groupe DF_i , où U_i est l'ensemble des attributs apparaissant dans DF_i .
4. Si aucun des schémas définis ne contient de clé X de R , rajouter un schéma $\langle R_{k+1}(X), \{\} \rangle$.

Remarque

- ◆ Certains schémas définis doivent être regroupés, lorsque l'ensemble d'attributs de l'un est inclus dans l'ensemble d'attributs de l'autre.

$R(A, B, C, D)$

$DF = \{A, B \rightarrow C,$

$C \rightarrow A,$

$B \rightarrow D\}$

Normalisation par décomposition

- On part du schéma initial et on le remplace par 2 autres schémas, contenant moins d'attributs et ainsi de suite jusqu'à n'avoir que des schémas dans la forme normale désirée.
- Formation d'un arbre de décomposition.

Algorithme decomp()

- Entrée : $\langle R(U), DF \rangle$
- Sortie : une décomposition
 $\langle R_1(U_1), DF|_{U_1} \rangle, \dots, \langle R_n(U_n), DF|_{U_n} \rangle$, uniquement spi où toutes les relations sont en FNBCK
- On crée un arbre de décomposition

cont.

1. Choisir une dépendance non triviale de DF $X \rightarrow Y$, X ne contenant pas de clé.
2. Décomposer en $R1(X, Y)$; $R2 = U - (Y - X)$
3. Pour $i = 1, 2$ faire
 - Si $\langle R_i, DF|_{R_i} \rangle$ est en FNBCK alors il devient feuille ;
 - Sinon $decomp(\langle R_i, DF|_{R_i} \rangle)$
4. Renvoyer l'ensemble de toutes les feuilles.

Remarque

- Les algorithmes par décomposition n'imposent pas de travailler avec la couverture minimale. D'où par exemple la précaution $R2 = U - (Y-X)$
- Dans le cas d'une couverture minimale, on mettrait juste $R2 = U - Y$.

Conclusion

- IL faut faire un compromis entre performance et absence de problèmes.

- Exemple : $R(A, B, C, D)$

$$DF = \{A,B \rightarrow C ; A,B \rightarrow D ; C \rightarrow A\}$$

est 3FN mais pas FNBCK.

- Une décomposition en FNBCK est-elle souhaitable sachant qu'elle génère 2 relations (et qu'il faudra faire des jointures pour retrouver certaines informations ?)