Bases de données

Chapitre 3 - Théorie des BD relationnelles

Source : Céline Rouveirol

CM3 du 28/09/2022

Représentation tabulaire d'une base de données

Aeroport	Id	Nom	Ville	Dept	CapaciteAn
	ORY	Orly	Orly	94	32 000 0000
	CDG	Charles de Gaulle	Roissy	93	43 500 000
	MRS	Provence	Marseille	13	1000000
	MTP	Fréjorgues	Montpellier	34	500000

Liaison	Id_Dép	Id_Arr
	ORY	MRS
	ORY	MTP
	CDG	MRS
	MRS	ORY
	MRS	CDG

Information

- explicite: Orly est dans le 94
- implicite: On peut aller du 94 au 13 en avion.

Schémas - Instances de relations

Aeroport	Id	Nom	Ville	Dept	CapaciteAn
	ORY	Orly	Orly	94	32 000 0000
	CDG	Charles de Gaulle	Roissy	93	43 500 000
	MRS	Provence	Marseille	13	1000000
	MTP	Fréjorgues	Montpellier	34	500000

Liaison	Id_Dép	Id_Arr
	ORY	MRS
	ORY	MTP
	CDG	MRS
	MRS	ORY
	MRS	CDG

Base de données

- 2 schémas de relations
  $Aeroport(Id, Nom, Ville, Dept, CapaciteAn)$ et
  $Liaison(Id\_Dep, Id\_Arr)$
- 2 tables ou instances de relation

Premières définitions (1)

Définition

Un *schéma de relation* est formé

1. d'un nom R (*Aeroport*, *Liaison*,...)
2. d'un ensemble d'attributs $Att(R)$
   ($Att(Aeroport) = (Id, Nom, Ville, Dept, CapaciteAn)$, ...)
3. pour chaque attribut $A \in Att(R)$, d'un domaine ou ensemble de valeurs $Dom(A)$
   ($Dom(Nom) =$ chaînes de caractères de longueur 20, ...)

- L'*arité* de R est le cardinal de $Att(R)$, la *cardinalité* d'une instance r de R est le nombre de n-uplets de l'instance.
- NB: deux relations différentes peuvent avoir le même schéma.

Premières définitions (2)

Définition

Un *schéma de base de données relationnelle* $\mathcal{R}$ est un ensemble (fini) de schémas de relations $\{R_1, \ldots, R_m\}$.

Définition

Soit la relation R de schéma $(A_1 : dom_1, \ldots, A_n : dom_n)$. Une *instance* r du schéma de R est un sous-ensemble fini (éventuellement vide) de n-uplets de $dom(A_1) \times dom(A_n)$.

Définition

Soit une base de données $\mathcal{R}$, une instance de $\mathcal{R}$ est un ensemble d'instances (éventuellement vides) pour chaque relation R_i $(1 \leq i \leq m)$ de $\mathcal{R}$.

Contraintes d'intégrité

Une *base de données* sur $\mathcal{R}$, $\mathcal{A}$ et $\mathcal{D}$ est définie par

- un schéma sur $\mathcal{R}$, $\mathcal{A}$
- une instance de son schéma
- un ensemble de contraintes d'intégrité sur ce schéma

Types de contraintes d'intégrité

1. Dépendances fonctionnelles (df): dépendance entre valeurs d'attributs d'une relation
   - $Employe(Nom, Prenom, Fonction, Service)$
     $Nom, Prenom \rightarrow Service$: le nom et le prénom d'une personne *déterminent* son service
2. Dépendances d'inclusion (di): dépendance entre valeurs d'attributs d'une ou deux relations
   - $Annuaire(Nom, Prenom) \subseteq Employe(Nom, Prenom)$: n'apparaissent dans l'annuaire que des employés de l'entreprise référencés dans *Employe*.
3. ...

Dépendances d'inclusion

- U ensemble d'attributs, $R[U]$: restriction du schéma R aux attributs U ($U \subseteq att(R)$). $t[U]$ est la valeur du n-uplet t "projeté" sur l'ensemble d'attributs U
- Une *dépendance d'inclusion* (di) indique une dépendance entre valeurs d'attributs de relation(s) différente(s)
- Portée : une di est définie sur deux relations R et S (R pouvant être égale à S).
- $R[A_1, \ldots, A_n] \subseteq S[B_1, \ldots, B_n]$ avec
  - $\{A_1, \ldots, A_n\} \subseteq att(R)$
  - $\{B_1, \ldots, B_n\} \subseteq att(S)$
  - pour tout $A_i \in R$, $B_i \in S$: $dom(A_i) \subseteq dom(Bi)$

Dépendances d'inclusion

- Sémantique intuitive: les valeurs de $A_1, \ldots, A_n$ de R sont des valeurs de $B_1, \ldots, B_n$ de S.

Définition

si $R[A_1, \ldots, A_n] \subseteq S[B_1, \ldots, B_n]$ est une *dépendance d'inclusion* définie sur deux relations R et S, alors pour toutes instances valide r, s de R et S

$$\text{Pour tout } t_1 \in r, \text{ il existe } t_2 \in s \text{ tq } t_1[A_1, \ldots, A_n] = t_2[B_1, \ldots, B_n]$$

Exemple

R	A	B	C
	a_1	b_1	c_1
	a_1	b_2	c_2
	a_2	b_3	c_3

S	A	B	D
	a_1	b_1	d_1
	a_1	b_2	d_2
	a_2	b_3	c_3
	a_3	b_4	c_4

$R[A, B] \subseteq S[A, B]$ est vérifiée
$S[A] \subseteq R[A]$ n'est pas vérifiée

Dépendance fonctionnelle

Définition

- Une dépendance fonctionnelle indique une dépendance entre *valeurs* d'une *même* relation R
- Sa portée est la relation R
- Une dépendance fonctionnelle se note $X \rightarrow Y$ et se lit X *détermine* Y
- X et Y sont des sous-ensembles d'attributs (éventuellement vides) tels que $X \cup Y \subseteq Att(R)$
- Intuition : la df $X \rightarrow Y$ associée à une relation R indique que tous les n-uplets de R qui ont la même valeur sur les attributs de X ont aussi la même valeur sur les attributs de Y.

Dépendance fonctionnelle

Définition

Définition

Si $X \rightarrow Y$ est une dépendance fonctionnelle définie sur une relation R alors toute instance valide r du schéma de R satisfait $X \rightarrow Y$:

$$\forall t_1, t_2 \in r \text{ tq } t_1[X] = t_2[X] \text{ alors } t_1[Y] = t_2[Y]$$

Exemple

Soit la relation $Employé(Nom, Prénom, Fonction, Service)$ et la dépendance fonctionnelle $Nom, Prénom \rightarrow Service$.
Une personne est associée à un seul service (mais peut avoir plusieurs fonctions dans ce service).

Dépendance fonctionnelle

Exemple

- Instance d'*Employé* satisfaisant la df *Nom*, *Prénom* $\rightarrow$ *Service*

Employé	Nom	Prénom	Fonction	Service
	Rouveirol	Céline	Enseignant	A^3
	Borne	Sylvie	Enseignant	AOC
	Rouveirol	Céline	Chercheur	A^3
	Borne	Sylvie	Chercheur	AOC
	...	...	...	...

- Instance d'*Employé* ne satisfaisant pas la df précédente

Employé	Nom	Prénom	Fonction	Service
	Rouveirol	Céline	Enseignant	A^3
	Borne	Sylvie	Enseignant	AOC
	Rouveirol	Céline	Chercheur	LCR
	Borne	Sylvie	Chercheur	AOC
	...	...	...	...

Fermeture d'un ensemble d'attributs

Définition

Soit R un schéma de relation, soit $\mathcal{F}$ un ensemble de dfs portant sur R, et $X \subseteq Y \subseteq Att(R)$.
Y est la *fermeture* de X dans R étant donné $\mathcal{F}$ si et ssi Y est le plus grand ensemble d'attributs tel que $X \rightarrow Y$.
On note la fermeture de X étant donné $\mathcal{F}$ par X^+.

```
1:  Fonction FermetureX(R, F et X)
2:      X+ := X
3:      Tant que X+ n'atteint pas un point fixe Faire
4:          Pour Tout f : Z → A ∈ F Faire
5:              Si Z ⊆ X+ Alors
6:                  X+ := X+ ∪ A
7:              Fin Si
8:          Fin Pour
9:      Fin Tant que
10:     Retourner X+
11: Fin Fonction
```

Clés et sur-clés

Le calcul de fermeture permet d'exhiber les clefs d'une relation.

- Idée: une clef de la relation R est un sous-ensemble minimal d'attributs X de R qui permet de distinguer deux n-uplets d'une instance r de R: $\forall t_1, t_2 \in r, t_1[X] \neq t_2[X]$.

Définition

Les sur-clefs d'une relation R avec son ensemble $\mathcal{F}$ de dfs sont les ensembles d'attributs $X \subseteq att(R)$ tels que $X^+ = att(R)$ ou $X \rightarrow att(R) \in F^+$.
X est une clef s'il n'existe pas de $Y \subset X$ tel que Y est une sur-clef.

Clés et sur-clés

Exemple

Soit la relation $Service(Nom, Division, Equipe)$ avec
$\mathcal{F} = \{Nom \rightarrow Division, Equipe \rightarrow Division, Equipe \rightarrow Nom\}$.

- $\{Equipe\}$ est une clef, car $\{Equipe\}^+ = \{Nom, Division, Equipe\}$.
- $\{Nom, Equipe\}$, $\{Division, Equipe\}$ et $\{Nom, Equipe, Division\}$ sont des sur-clefs.

Clé étrangère

- Une clé étrangère est une dépendance d'inclusion $R(X) \subseteq S(Y)$ telle que Y est la clé primaire de S.

Exemple

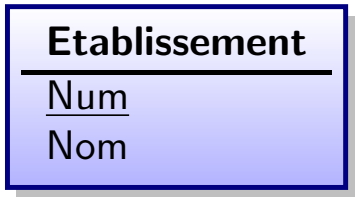
Soit la relation $Service(Nom, Division, Equipe)$ avec $\mathcal{F} = \{Nom \rightarrow Division, Equipe \rightarrow Division, Equipe \rightarrow Nom\}$ et la relation $Employe(NomE, PrenomE, Equipe)$ avec le di $Employe(Equipe) \subseteq Service(Equipe)$. $\{Equipe\}$ est une clé de $Service$ et $Equipe$ est une clé étrangère de $Employe$.

Passage du modèle E/A au modèle relationnel

- On passe d'un modèle disposant de deux structures (entité - association) à un modèle ne disposant que d'une seule structure (la relation)
- On applique un ensemble de règles qui garantissent la cohérence sémantique entre le modèle E/A et le modèle relationnel

Entités

- Chaque entité du modèle E/A devient une relation de même nom, avec les mêmes attributs que l'entité
- L'identifiant de l'entité devient la clef de la relation
- Les attributs n'appartenant pas à la clef ne doivent être en df qu'avec la clef, et toute la clef.



modèle E/A

$\Rightarrow$

etablissement(num,nom)

modèle relationnel

- Dans un nuplet d'une table une valeur NULL représente une valeur indéterminée
- Impossible d'avoir des valeurs NULL dans les clefs !!!

Clés étrangères

- Soient R et S deux relations. L'ensemble d'attributs X de R forme une clef étrangère de R ssi il existe une contrainte d'inclusion $R[X] \subseteq S[Y]$ telle que Y est la clef de S.
- On peut avoir des valeur NULL dans les clefs étrangères !!!

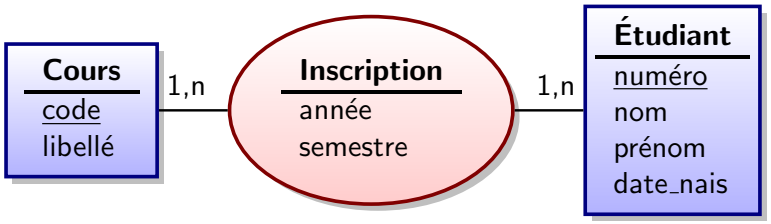
Exemple

Batiment(<u>numero</u>,adresse)
SalleDeCours(<u>nom</u>,capacite,equipement,numero)
Salle de cours[numero] $\subseteq$ Batiment[numero]

Associations

Association de type n-n

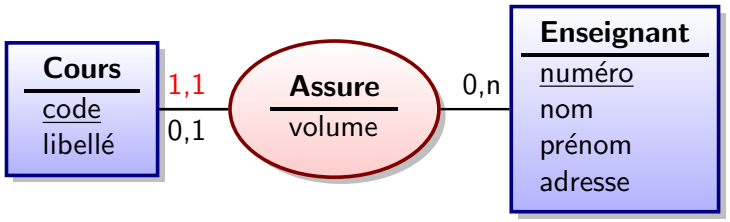
- l'association du modèle E/A devient une relation de même nom, avec les mêmes attributs que ceux de l'association
- la clef est la concaténation des clefs de chaque relation qui sont aussi des clefs étrangères

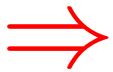


Cours(code, libellé)
Etudiant(numéro,nom,prénom,date_naiss)
Inscription(numéro,code,année,semestre)
Inscription[numéro] $\subseteq$ étudiant[numéro]
Inscription[code] $\subseteq$ cours[code]

Association de type 1-n

- La clef côté *n* devient la clef étrangère côté 1
- Les attributs de l'association vont côté 1

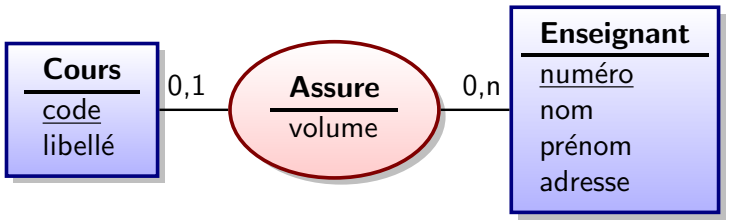




1,1 Enseignant(<u>numéro</u>,nom,prénom,adresse)
1,1 Cours(<u>code</u>,libellé,numéro NOT NULL, volume)
0,1 Cours(<u>code</u>,libellé,numéro, volume)
Cours[numéro] $\subseteq$ Enseignant[numéro]

Association de type 1-n SANS NULL

- On fait de l'association une relation
- Les clefs des entités sont les clefs (étrangères), la clé de la relation codant l'association est la clé côté 1.



Enseignant(<u>numéro</u>,nom,prénom,adresse)
Cours(<u>code</u>,libellé)
Assurer(numéro,<u>code</u>,volume)
Assurer[numéro] $\subseteq$ Enseignant[numéro]
Assurer[code] $\subseteq$ Cours[code]

Entités faibles

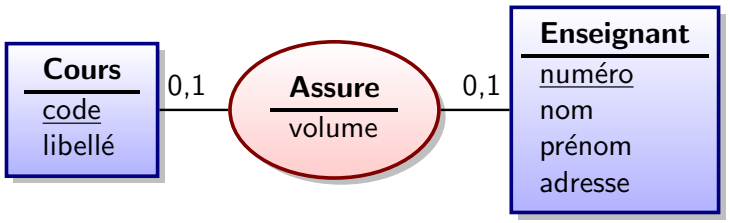
- La clef côté *n* devient
  - la clef avec l'identifiant de l'entité faible
  - une clef étrangère côté 1

$\Rightarrow$

Bâtiment(<u>numéro</u>,adresse)
Salle de cours(<u>no</u>,capacité,équipt,<u>numéro</u>)
Salle de cours[numéro] $\subseteq$ Bâtiment[numéro]

Association de type 1-1

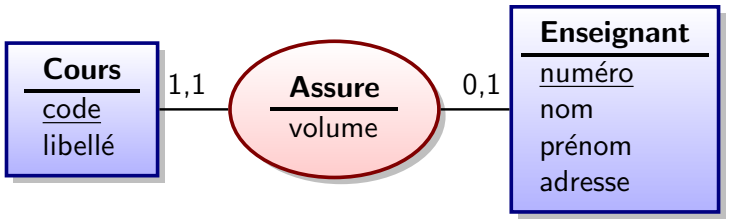
- Si les 2 cardinalités minimales sont 1, on peut fusionner les 2 entités
- Si les 2 cardinalités minimales sont à 0
  - Les clefs deviennent clefs étrangères
  - Les clefs étrangères peuvent prendre la valeur NULL



AVEC NULL
Enseignant(numéro,nom,prénom,adresse,code)
Cours(code,libellé)
Enseignant[code] $\subseteq$ Cours[code]
OU
Enseignant(numéro,nom,prénom,adresse)
Cours(code,libellé,numéro)
Cours[numéro] $\subseteq$ Enseignant[numéro]

Association de type 1-1

- 1 cardinalité minimale à 0, sans NULL

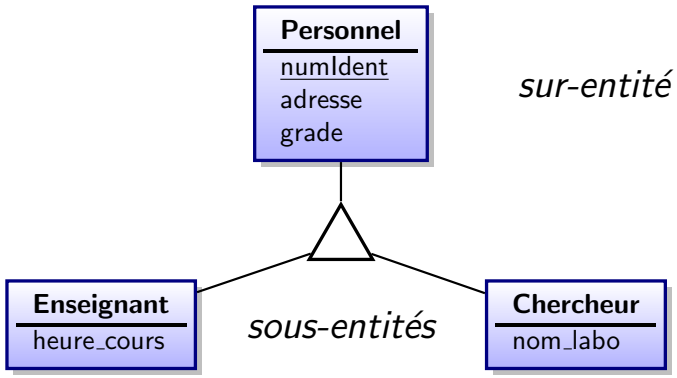


$\Rightarrow$

Enseignant(numéro,nom,prénom,adresse)
Cours(code,libellé)
Assurer(numéro,code)
Assurer[numéro] $\subseteq$ Enseignant[numéro]
Assurer[code] $\subseteq$ Cours[code]
ou
Enseignant(numéro,nom,prénom,adresse)
Cours(code,libellé,numéro NOT NULL)
Cours[numéro] $\subseteq$ enseignant[numéro]

Association d'héritage

- Chaque sous-entité est transformée en une relation; la clef primaire de la sur-entité migre dans les relations issues des sous-entités et devient clef et clef étrangère



$\Rightarrow$

Personnel(numIdent, adresse, grade)
Enseignant(numIdent, HeureCours)
Chercheur(numIdent, NomLabo)
Enseignant[numIdent] $\subseteq$ Personnel[numIdent]
Chercheur[numIdent] $\subseteq$ Personnel[numIdent]

Dépendances fonctionnelles impliquées (1)

- Toutes les dépendances fonctionnelles ne sont pas explicites !
- Certaines dfs sont impliquées par d'autres dfs.
- Exhiber ces dfs lors de la modélisation :
  - calcul des clefs des relations
  - découvrir que les dfs définies *a priori* pour une application à développer impliquent des dfs non souhaitées pour cette application
  - découvrir des dfs redondantes
  - ...
- Exhiber ces dfs lors de la conception :
  - améliorer le schéma d'une base de données par normalisation (cf. partie du cours sur la normalisation)
  - ...

Dépendances fonctionnelles impliquées (2)

- Il existe un système qui permet de calculer toutes les dépendances impliquées d'un ensemble de dfs: le système d'Armstrong
- Les trois axiomes d'Armstrong sont :
  1. Reflexivité: si $Y \subseteq X$ alors $X \rightarrow Y$
  2. Augmentation: si $X \rightarrow Y$ alors $XZ \rightarrow YZ$ pour tout Z
  3. Transitivité: si $X \rightarrow Y$ et $Y \rightarrow Z$ alors $X \rightarrow Z$
- Des règles additionnelles se déduisent des précédentes :
  1. Union: si $X \rightarrow Y$ et $X \rightarrow Z$ alors $X \rightarrow YZ$
  2. Décomposition: si $X \rightarrow YZ$ alors $X \rightarrow Y$ et $X \rightarrow Z$
  3. Pseudo-transitivité: si $X \rightarrow Y$ et $YZ \rightarrow W$ alors $XZ \rightarrow W$
  4. Augmentation àx gauche: si $X \rightarrow Y$ alors $XZ \rightarrow Y$
- Le système d'Armstrong est correct et complet.

Dépendances fonctionnelles impliquées (3)

- Les axiomes d'Armstrong traduisent effectivement la sémantique des dfs.

Proof.

Preuve pour l'augmentation. Soient R une relation et F son ensemble de dfs associé contenant la df $X \rightarrow Y$. Soit r une instance valide de R ($r \models F$). Soient t_1 et t_2 deux n-uplets de r tels que $t_1[XZ] = t_2[XZ]$.
On cherche à démontrer que $t_1[YZ] = t_2[YZ]$.
Comme $t_1[XZ] = t_2[XZ]$ et $X \rightarrow Y$, et par définition des dfs, on a $t_1[XYZ] = t_2[XYZ]$, et donc $t_1[YZ] = t_2[YZ]$. □

Dépendances fonctionnelles impliquées (4)

- On retrouve les règles additionnelles par dérivation à partir des axiomes d'Armstrong

Proof.

Pour l'union: si $X \to Y$ et $X \to Z$ alors $X \to YZ$

(1)	$X \to Y$	donné
(2)	$X \to XY$	augmentation de (1)
(3)	$X \to Z$	donné
(4)	$XY \to YZ$	augmentation de (3)
(5)	$X \to YZ$	transitivité de (2) et (4)

Dépendances fonctionnelles impliquées et fermeture

Définition

La *fermeture* d'un ensemble $\mathcal{F}$ de dépendances fonctionnelles (notée $\mathcal{F}^+$) d'une relation R est telle que $\mathcal{F} \models X \rightarrow Y$ ssi il existe $X \rightarrow X^+ \in \mathcal{F}^+$ avec $Y \subseteq X^+$

Algorithme de calcul de fermeture d'un ensemble de dfs

1: Fonction *FermetureDf*(R, $\mathcal{F}$)
2: Retourner $\cup_{\forall X \subseteq Att(R)}\{X \rightarrow X^+\}$
3: Fin Fonction

Fermeture d'un ensemble de dfs

Exemple

Soit la relation $Service(Nom, Division, Equipe)$ avec
$\mathcal{F} = \{Nom \rightarrow Division; Equipe \rightarrow Division; Equipe \rightarrow Nom\}$.

$Nom^+ = Nom, Division$
$Division^+ = Division$
$Equipe^+ = Equipe, Nom, Division$

$$\begin{aligned}\mathcal{F}^+ = \{ \quad & Nom \rightarrow Nom, Division; Division \rightarrow Division; \\ & Equipe \rightarrow Equipe, Division, Nom; \\ & Nom, Division \rightarrow Nom, Division; \\ & Nom, Equipe \rightarrow Nom, Equipe, Division; \\ & Division, Equipe \rightarrow Division, Equipe, Nom; \\ & Nom, Equipe, Division \rightarrow Nom, Equipe, Division\}\end{aligned}$$

Dépendances fonctionnelles

Redondance

- La fermeture d'un ensemble de dfs permet de calculer des ensembles de dfs *minimaux*.
- Idée: on cherche les plus petits sous-ensembles F' d'un ensemble F de dfs tels que:
  - les dépendances de $F \setminus F'$, (explicites dans F), sont impliquées (implicites) dans F'
  - les dfs de F' comportent le moins d'attributs possibles

Dépendances fonctionnelles

Redondance

Définition

Soit F un ensemble de dfs défini sur une relation R. F est minimal ssi pour tout $F' \subset F$ $F'^{+} \neq F^{+}$ avec :

- toutes les df de F' ont un seul attribut à droite
- si $X \rightarrow Y$ est dans F' alors il n'existe pas $X' \rightarrow Y$ dans F' avec $X' \subset X$

Exemple

Soit la relation $Service(Nom, Division, Equipe)$ avec l'ensemble de dfs $\mathcal{F} = \{Nom \rightarrow Division; Equipe \rightarrow Division; Equipe \rightarrow Nom\}$.
$F_{min} = \{Nom \rightarrow Division; Equipe \rightarrow Nom\}$.

Algorithme de calcul d'UN ensemble de dfs minimal

1: Fonction *DFMinimal*(R, $\mathcal{F}$)
2: $\mathcal{F}' := \mathcal{F}$ ▷ *Réduction à droite des dfs de* $\mathcal{F}'$
3: Pour Tout $f : X \rightarrow A_1, \ldots, A_n \in \mathcal{F}'$ Faire
4: Remplacer f dans $\mathcal{F}'$ par $\{X \rightarrow A_1, \ldots, X \rightarrow A_n\}$
5: Fin Pour
6: ▷ *Réduction à gauche des dfs de* $\mathcal{F}'$
7: Pour Tout $f : X \rightarrow A_i \in \mathcal{F}'$ Faire
8: Remplacer f par $X' \rightarrow A_i$ si $X' \subset X$ et $A_i \in X'^+$ et qu'il n'existe pas de X'' tel que $X'' \subset X'$ et $A_i \subset X''^+$
9: Fin Pour
10: ▷ *Réduction des dfs de* $\mathcal{F}'$
11: Pour Tout $f : X \rightarrow A_i \in \mathcal{F}'$ Faire
12: Si $\mathcal{F} \setminus \{X \rightarrow A_i\} \models X \rightarrow A_i$ Alors Supprimer f
13: Fin Si
14: Fin Pour
15: Retourner $\mathcal{F}'$
16: Fin Fonction

Analyse de l'algorithme

- Les étapes 2 et 3 font qu'il existe, dans le cas général, plusieurs ensembles minimaux bien qu'un seul soit réellement calculé. Un ensemble minimal est aussi appelé *couverture minimale*.
- Pour un ensemble de dfs (sauf indication contraire):
  - les dfs triviales sont omises: si l'ensemble contient $X \rightarrow Y$ avec $Y \subseteq X$ alors $X \rightarrow Y$ est omise
  - ses dfs trivialement redondantes sont omises: si l'ensemble contient $X \rightarrow Y$ et $T \rightarrow U$ avec $T \subseteq X$ et $Y \subseteq U$ alors $X \rightarrow Y$ est omise

Conclusion sur les dfs

- Les dfs jouent un rôle important dans la modélisation d'une base de données. Elles permettent de contraindre une relation: elles restreignent les instances possibles de son schéma en éliminant des instances non valides pour une application donnée.
- Le mécanisme de fermeture des df permet :
  - de vérifier que les dfs associées à une relation sont appropriées (i.e. que les dfs impliquées d'une relation modélisent bien l'application)
  - de minimiser l'ensemble des dfs d'une relation
  - de trouver les clefs possibles pour accéder aux nuplets d'une instance de schéma de relation.
  - de transformer une modélisation afin que sa mise en oeuvre débouche sur une base limitant (voir supprimant) la redondance de données et les anomalies de mise-à-jour.