

# Bases de données

## *Conception*

### Intégrité, redondance et normalisation (version intégrale)

MLR\_01  
v271a

2024-05-12

Christina.Khnaisser@USherbrooke.ca

Luc.Lavoie@USherbrooke.ca

© 2018-2021, Μητίς (<http://info.usherbrooke.ca/llavoie>)

CC BY-NC-SA 4.0 (<https://creativecommons.org/licenses/by-nc-sa/4.0/>)

# Plan

- **Normalisation**
  - Définitions et concepts
- **Représentation normalisée**
  - Atomicité
  - 1FN
- **Dépendances fonctionnelles**
  - Définition
  - FNBC
  - Théorème de Heath
- *Autres dépendances fonctionnelles*
  - 3FN
  - 2FN
  - Comparaison 2FN, 3FN, FNBC
- **Dépendances de jointure**
  - 5FN et théorème général de projection-jointure
  - 4FN et théorème de Fagin
  - 6FN, l'ultime frontière?
  - Les algorithmes

## **Note**

*les sections dont le titre est en gris italique peuvent être omises en première lecture.*

# Normalisation

- Rappels
- Redondance
- Définitions
- Formes de normalisation
- Portée du module
- Vision traditionnelle
- Vision contemporaine
- Pourquoi réduire la redondance?
- Comment réduire la redondance?
- Le rôle des contraintes
- Pourquoi ne pas se contenter de contraintes?

## Normalisation

### Rappels

- Pour être *adéquat*, un modèle logique de BD doit minimalement être cohérent, valide et efficace.
- En pratique, il doit être également être évolutif, efficient et aussi complet que possible.
- Un modèle logique est *complet* en regard d'un modèle conceptuel si chacun des prédicats exprimables au sein du modèle conceptuel y est représentable sous la forme d'une variable de relation (de base ou virtuelle).

## Normalisation

### Redondance

- Si la complétude est souhaitable, voire nécessaire, elle ne doit pas être obtenue au prix de la redondance, car celle-ci entraîne souvent des problèmes d'efficacité, d'évolutivité et d'efficience. Elle peut même favoriser l'apparition de l'incohérence et de l'invalidité.
- En effet, lorsque la redondance des données entraîne la redondance des propositions, elle devient un facteur important d'incohérences potentielles lors de
  - la mise à jour des données,
  - l'interrogation des données,
  - l'évolution du modèle logique.
- La réduction de la redondance et, à défaut de son élimination, son contrôle est donc un objectif majeur de qualité.

## Normalisation

### Définitions

- En théorie relationnelle, une *forme normale* désigne un critère d'évaluation de la redondance (et donc de la non-redondance).
- Par extension, la *normalisation* désigne une transformation applicable à un MLD permettant à celui-ci de s'approcher ou d'atteindre une forme normale.
- En outre, une normalisation adéquate doit maintenir la complétude du MLD en regard du MCD.

## Normalisation

### Formes de normalisation

- On distingue plusieurs catégories de normalisation de modèles logiques relationnels selon la perspective utilisée, par exemple :
  - des attributs (1FN)
    - assure que tout attribut est associé à exactement une valeur de son type de définition;
  - des propositions/prédicats (FNBC, 5FN, 6FN)
    - assure que si une même dépendance est représentée plus d'une fois dans la base de données, toutes ses représentations sont cohérentes;
  - des intervalles (FNI)
    - assure l'unicité, l'intégrité et la cohérence d'une relation compacte en regard d'attributs clés de type intervalle;
  - historique (FNBT)
    - assure la cohérence historique des prédicats bi-temporalisés.

## Normalisation

### Parallèle mathématique

- À strictement parler, seule la 1FN traite de normalisation au sens mathématique usuel, à savoir *une représentation biunivoque de toute relation respectant les axiomes de la théorie relationnelle*.
- Toutes les autres «formes normales» relationnelles sont en fait des critères qui aident à réduire, voire éliminer, la redondance donc à concevoir un modèle logique cohérent, complet et évolutif).



## Normalisation

### Portée du module

- Le module s'intéresse aux normalisations d'attributs et de propositions.
- Plus spécifiquement
  - Normalisation des attributs : 1FN
  - Normalisation des propositions : 2FN, 3FN, FNBC, 4FN, 5FN, 6FN
- Cependant, comme
  - $6FN \Rightarrow 5FN \Rightarrow 4FN \Rightarrow FNBC \Rightarrow 3FN \Rightarrow 2FN \Rightarrow 1FN$
  - 4FN et 3FN ne sont utiles qu'à des fins d'illustration
  - 2FN est inutile en pratique
- Le module se concentre sur les formes essentielles
  - 1FN, FNBC, 5FN et 6FN

## Normalisation

### Vision

- La conception relationnelle vise à donner une garantie *structurelle* de cohérence.
- Les moyens retenus pour atteindre ce but sont (d'abord) la minimisation de la redondance et (ensuite) l'ajout de contraintes.
- Les formes normales sont donc des moyens d'appliquer un principe de conception dans le but de minimiser la redondance.

## Normalisation

### Introspection

- Bien sûr, la cohérence n'est pas le seul principe applicable.
- Bien sûr, appliquer un moyen (la réduction de la redondance) sur un modèle logique incorrect a peu de chance de le corriger.
- La normalisation offre cependant l'opportunité de corriger certaines erreurs de conception en les rendant plus manifestes.

# Normalisation en regard des attributs (1FN)

- Atomicité
- Définitions
- Corolaires
- Observations
- Exemple
- Discussion

## Rappel TRM\_01: Fondements — Tuples

○ Soit  $a_i$  des identifiants distincts et  $D_j$  des types, un tuple  $t$  est défini comme suit:

- $t \triangleq (\{a_1:D_1, a_2:D_2, \dots, a_n:D_n\}; \{(a_1, v_1), (a_2, v_2), \dots, (a_n, v_n)\})$
- avec  $\forall i: 1 \leq i \leq \text{deg}(t) \Rightarrow \text{val}(t, a_i) \in \text{def}(t, a_i)$

○ où

- $\text{def}(t) = \{a_1:D_1, a_2:D_2, \dots, a_n:D_n\}$  entête de  $t$
- $\text{def}(t, a_i) = D_i$  type de  $a_i$  de  $t$
- $\text{val}(t) = \{(a_1, v_1), (a_2, v_2), \dots, (a_n, v_n)\}$  valeur de  $t$
- $\text{val}(t, a_i) = v_i$  valeur de  $a_i$  de  $t$
- $\text{deg}(t) = n$  degré de  $t$
- $\text{id}(t) = \{a_1, a_2, \dots, a_n\}$  les identifiants d'attributs de  $t$

## Atomicité

### Définition

- Un modèle logique relationnel est normalisé en regard des attributs si et seulement si toutes ses relations sont normalisées.
- Une relation E est normalisée en regard des attributs si et seulement si tous ses attributs sont atomiques.
- **Un attribut est atomique si et seulement s'il est associé à une et une seule valeur de son type de définition.**

## 1FN

### Définitions

- Soit  $r$  une variable de type relationnel  $R(a_1:T_1, \dots, a_n:T_n)$ ,  $r$  est normalisée en regard des attributs si et seulement si pour tout tuple  $t$  de  $r$ , pour tout  $1 \leq i \leq n$ , la valeur de l'attribut  $a_i$  de  $t$  est de type  $T_i$ .
- On désigne un modèle logique relationnel normalisé en regard des attributs comme étant en «*première forme normale*» (1FN).

## 1FN

### Corolaires

- **Un attribut ne peut être sans valeur**  
*(pas de marqueur NULL; l'extension de tous les types en une structure de treillis comprenant un supremum et un infimum demeure toutefois envisageable).*
- **La valeur d'un attribut est unique**  
*(pas d'ensemble de valeurs... à moins que l'attribut ne soit d'un type défini comme un ensemble d'«ensembles de valeurs» — auquel cas, l'«ensemble de valeurs» est une valeur atomique, scalaire).*
- **Toute relation est en 1FN par définition**  
*(voir le module TRM\_01 — Fondements).*



- *La théorie relationnelle permet l'existence d'attributs non scalaires (tuple, relation) dans la mesure où les constructeurs de types correspondants sont pris en charge par le modèle relationnel de référence.*
- *En pratique, les types et les constructeurs de types acceptés dépendent du SGBDR.*

## 1FN

### Transformations

- Si un attribut **non clé**  $x:T$  d'une relation  $E1$  n'est pas en 1FN :
  - Le retirer de  $E1$ .
  - Ajouter une nouvelle relation **totale**  $E2$  dont les attributs sont :
    - les attributs d'une clé candidate de  $E1$  (préférentiellement la clé primaire) ;
    - l'attribut  $x:T$ .
- Si un attribut **clé**  $a:T$  d'une relation  $E1$  n'est pas en 1FN :
  - Le remplacer par un attribut  $k:S$   
(où  $S$  est un type scalaire de cardinalité suffisante).
  - Ajouter une nouvelle relation **totale**  $E2$  dont les attributs sont :
    - l'attribut  $k:S$  ;
    - l'attribut  $a:T$ .

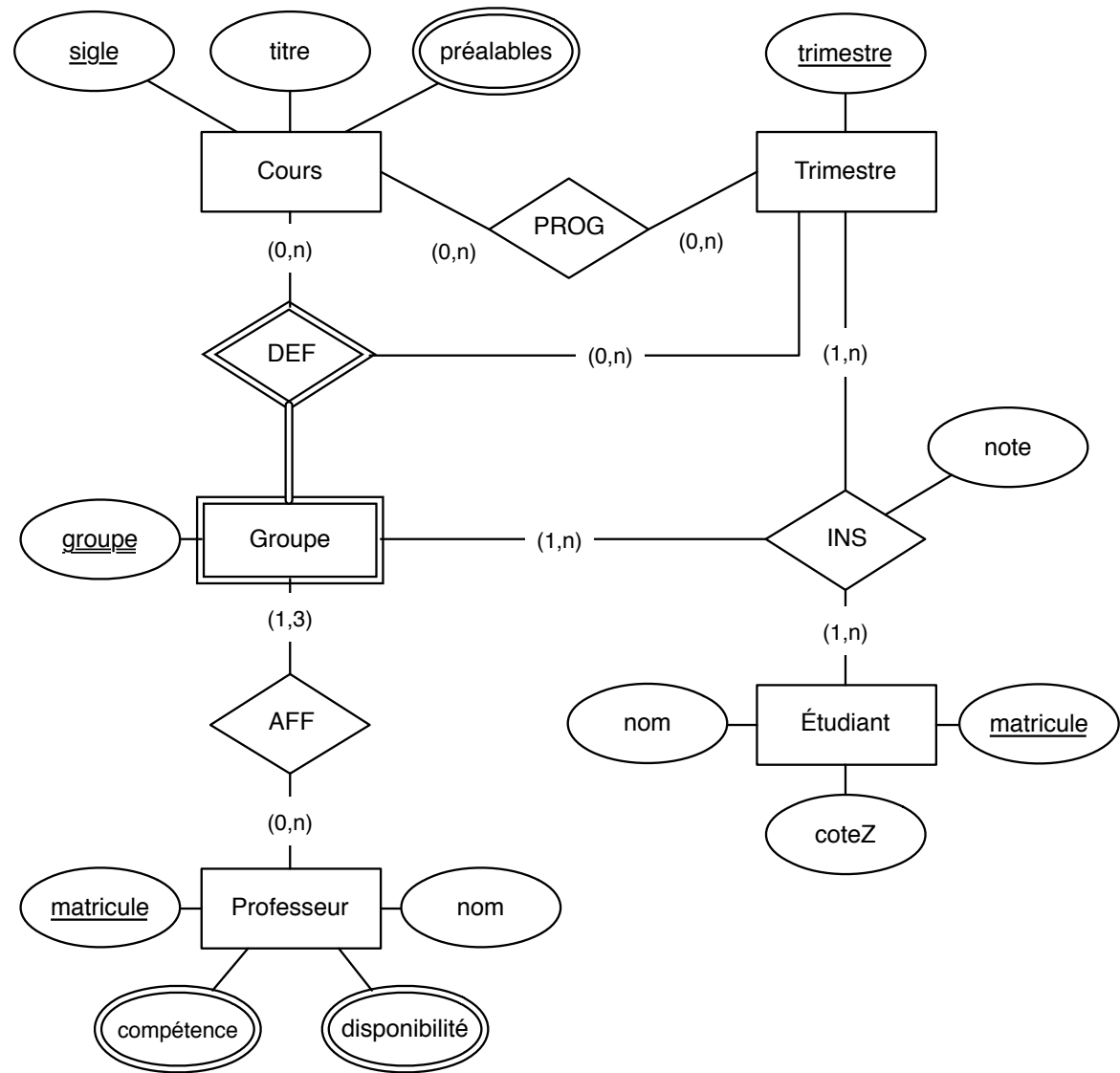
## 1FN – Exemple

### Exemples et contre-exemples de normalisation

- Les relations GroupeCours, Inscription, Étudiant, et Accessibilité sont en 1FN.
- La relation Cours n'est pas en 1FN, car l'attribut «préalables» est un ensemble de sigles.
- La relation Professeur n'est pas en 1FN, car les attributs «compétence» et «disponibilité» sont des ensembles.

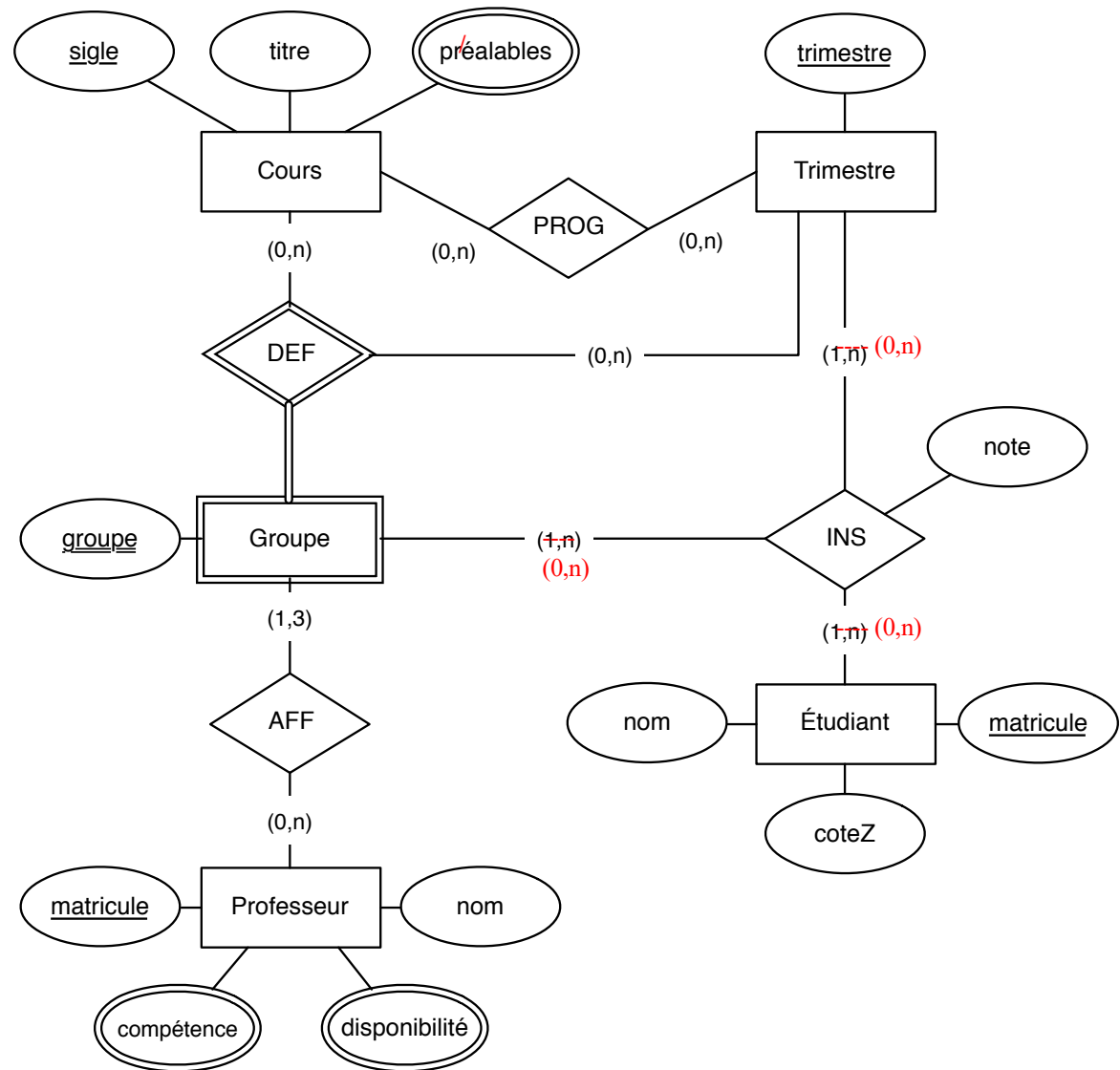
## 1FN – Exemple

### Le MCD des cours



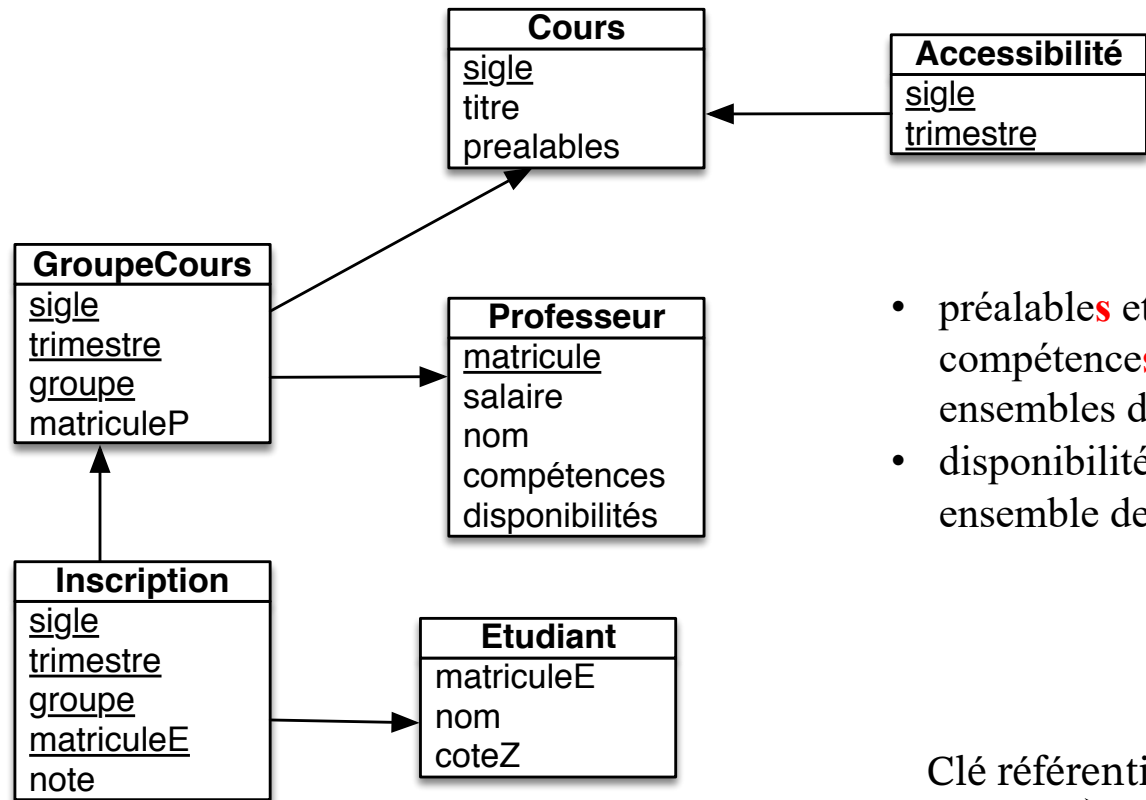
# 1FN – Exemple

## Le MCD des cours — corrigé



## 1FN – Exemple

### Le modèle logique relationnel «naïf» (avant normalisation)

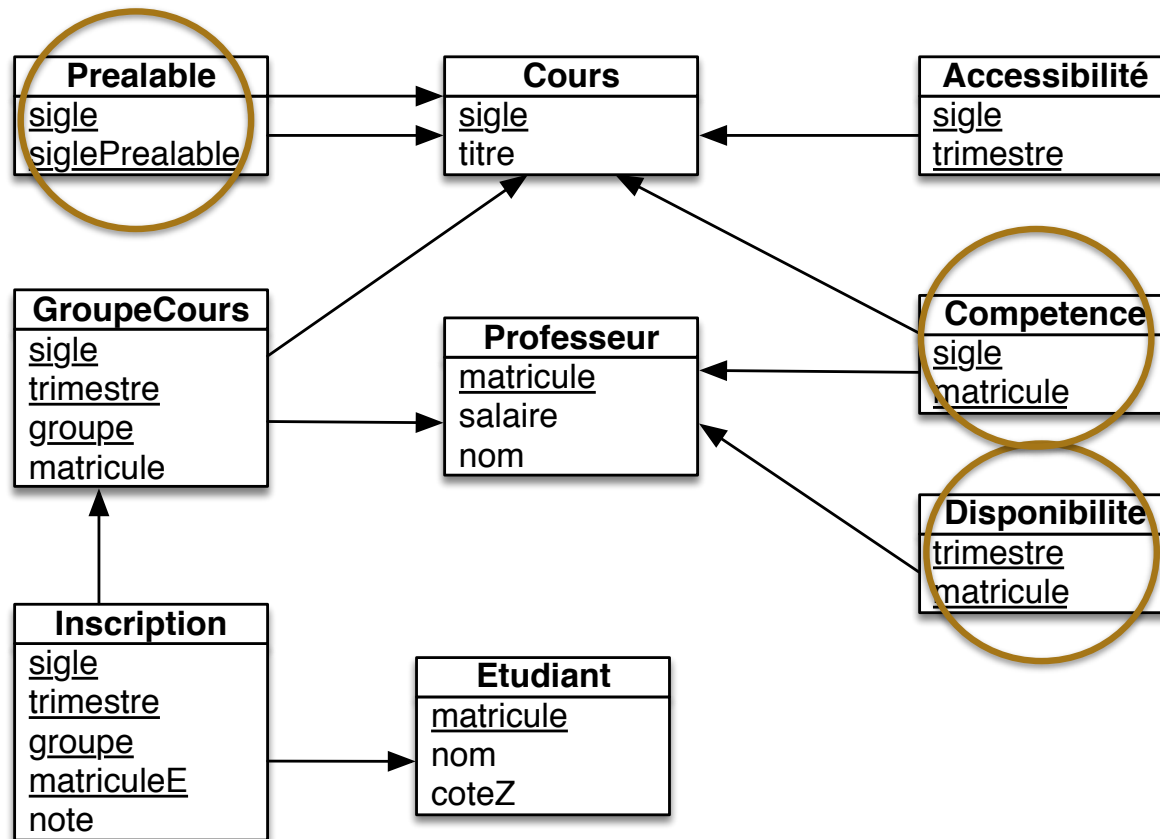


- préalables et compétences sont des ensembles de cours;
- disponibilité est un ensemble de trimestres;

Clé référentielle  
→

## 1FN – Exemple

### Le modèle logique des cours, après normalisation



## 1FN

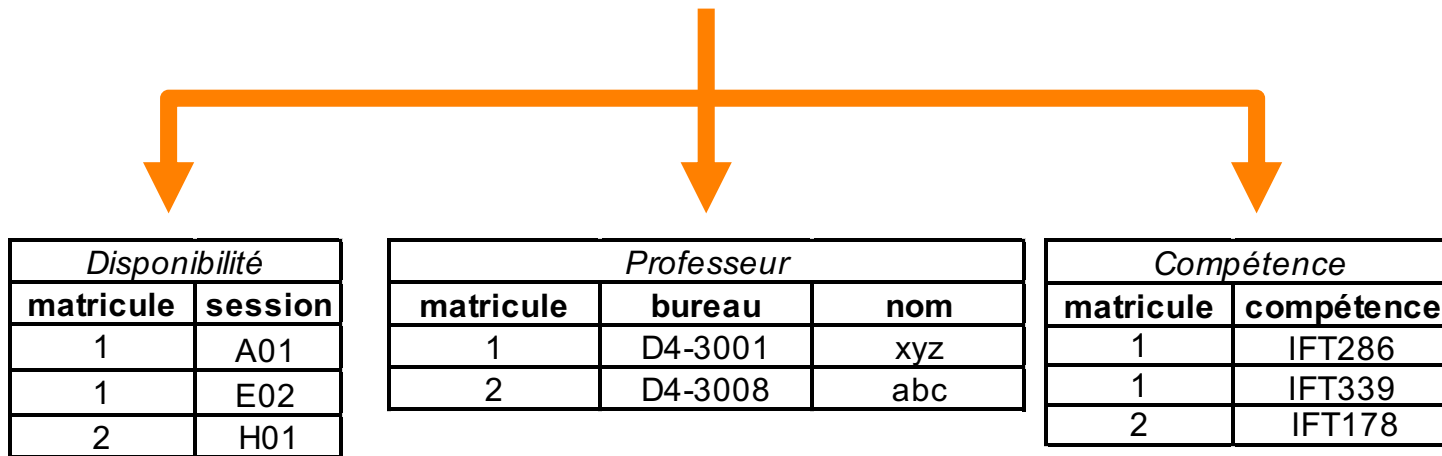
### Multiplicité cachée

- Une erreur fréquente consiste à définir le type d'un attribut comme un texte et d'y encoder plusieurs valeurs.



## 1FN – Exemple de normalisation d'attributs non clés

<i>professeur</i>				
matricule	bureau	nom	disponibilités	compétences
1	D4-3001	xyz	A01 E02	IFT286 IFT339
2	D4-3008	abc	H01	IFT178



## 1FN

### Pourquoi normaliser?

- La théorie relationnelle repose sur une prémisse de base : *toute variable est relationnelle et les opérateurs relationnels sont les seuls à permettre la composition des valeurs relationnelles (et qu'ils sont suffisants pour ce faire).*
- C'est grâce à cette prémisse, il est possible de garantir que tout prédicat relationnel (requête) peut être évalué tout en maintenant l'interprétation des relations comme prédicats logiques (des tuples comme propositions logiques).
- Cette fermeture des expressions relationnelles garantit également l'exactitude de toute substitution de deux expressions équivalentes l'une par l'autre et fournit une base solide à l'optimisation des requêtes.

## 1FN

### Pourquoi normaliser?

- La problématique de représentation liée au stockage d'une valeur non scalaire dans une BD (et la perte de performance qui en pourrait en découler) est souvent évoquée.
- Bien que souvent avérée, elle apparaît secondaire en regard de la cohérence, de la validité et de l'efficacité garanties par la 1FN.

- Certains modèles relationnels permettent néanmoins de définir des attributs à l'aide de types non scalaires :
  - **en maintenant la prémisse de base**, le modèle relationnel étendu (Tutorial D) autorise les attributs de type relationnel (opérateurs wrap et unwrap, group et ungroup);
  - **en levant (localement) la prémisse de base**, le modèle relationnel-objet (les multiset de ISO 9075:1999).

# Dépendances fonctionnelles

- Préambule
- Principe
- Définitions
  - Dépendance fonctionnelle
  - Précisions
  - Irréductibilité et trivialité
  - Clé candidate
- Axiomes d'Armstrong
- Représentation graphique
- Provenance des DF
- FNBC
- Théorème de Heath

## Préambule

### Exemple (1)

- Quelle est le redondance cachée dans cette table ?


Code	No	Titre	Crédit	Département
IFT	187	Éléments de bases de données données	3	Informatique
IMN	117	Acquisition des médias numériques	3	Informatique
IGL	301	Spécification et vérification des exigences	3	Informatique
IFT	697	Projet d'intégration et de recherche	6	Informatique
BIO	101	Biométrie	3	Biologie
PHY	131	Optique	2	Physique

## Préambule

### Exemple (2)

- La dépendance Code -> Département induit une redondance

Code	No	Titre	Crédit	Département
IFT	187	Éléments de bases de données données	3	Informatique
IMN	117	Acquisition des médias numériques	3	Informatique
IGL	301	Spécification et vérification des exigences	3	Informatique
IFT	697	Projet d'intégration et de recherche	6	Informatique
BIO	101	Biométrie	3	Biologie
PHY	131	Optique	2	Physique



Code	No	Titre	Crédit
IFT	187	Éléments de bases de données données	3
IMN	117	Acquisition des médias numériques	3
IGL	301	Spécification et vérification des exigences	3
IFT	697	Projet d'intégration et de recherche	6
BIO	101	Biométrie	3
PHY	131	Optique	2

Code	Département
IFT	Informatique
IMN	Informatique
IGL	Informatique
BIO	Biologie
PHY	Physique

## Préambule

### PRINCIPE

- En décomposant la relation en projections choisies de telle sorte à faire disparaître les redondances et dont la jointure est égale à la relation d'origine, c'est-à-dire :
- Soit  $E = \{a_1, \dots, a_n\}$  l'entête de  $R$ , choisir une décomposition  $D = \{d_1, \dots, d_k\}$  telle que
  1.  $d_1 \subseteq E, \dots, d_k \subseteq E$
  2.  $R = (R \pi d_1) \bowtie \dots \bowtie (R \pi d_k)$
  3. le nombre de redondances dans  $D$  soit le plus petit possible
  4. s'il existe encore des redondances, les contraindre de façon à ce qu'elle soit cohérente, au mieux identique.
- Toute la question se résume donc à choisir la bonne décomposition, c'est-à-dire le bon  $D$ .



## Préambule

### Persistance de redondances

- En faisant coïncider les dépendances fonctionnelles avec les décomposition, on réduit la reondance.
- Malheureusement, la décomposition n'est pas toujours possible, ni toujours efficiente.
- Des redondances peuvent donc persister dans un modèle logique.

## Préambule

### Comment contrôler la redondance résiduelle?

- Si une même proposition est stockée en plusieurs endroits, il faut (pour maintenir la cohérence) s'assurer qu'elle est la même partout:
  - ⇒ que les données qui la représentent soient les mêmes partout;
  - ⇒ que toute modification d'une donnée d'un emplacement est reflétée dans tous les autres emplacements;
  - ⇒ que tout retrait de la proposition dans un emplacement entraîne le retrait dans les autres emplacements.
- Pour contrôler les occurrences redondantes, on peut, *il faut*, les lier par une contrainte.

## Préambule

### Historique

- Historiquement, les formes normales et les algorithmes qui leur sont associés ont été «découverts» en ordre progressif, de la moins complète à la plus complète.
- Une première famille a été construite sur la décomposition binaire des relations fondées sur les seules dépendances fonctionnelles et culmine dans la forme normale de Boyce-Codd ( $2FN \Rightarrow 3FN \Rightarrow FNBC$ ).
- Une deuxième famille a été construite à partir de la constatation que certaines redondances ne pouvaient être éliminées par décomposition binaire et que des décompositions de degré supérieur pouvaient être requises, elle est fondée sur la notion plus fondamentale de dépendance de jointure et culmine dans la cinquième forme normale ( $4FN \Rightarrow 5FN$ ).
- Plusieurs autres formes normales ont été développées, dont la 6FN qui garantit l'indépendance des prédicats élémentaires.

## Définitions

### Dépendance fonctionnelle

- Un attribut A dépend fonctionnellement d'un ensemble d'autres attributs X s'il existe une fonction permettant de calculer le premier à partir des derniers.
  - $(A \text{ dépend fonctionnellement de } X) \equiv (X \text{ détermine fonctionnellement } A)$
  - X est le déterminant et A le déterminé.

## Définitions

### Notation

- Une dépendance fonctionnelle
  - $DF [X \rightarrow A]$  ou
  - $A_1, \dots, A_n \rightarrow A_{n+1}$  lorsque  $A_{n+1}$  dépend de  $X = \{A_1, \dots, A_n\}$
- Un ensemble de  $k$  DF ayant le même déterminant peut être dénoté ainsi:
  - $A_1, \dots, A_n \rightarrow A_{n+1}, \dots, A_{n+k}$

## Définitions

### Discussion (1)

- Il y a un lien entre dépendance fonctionnelle et redondance à partir du moment où la valeur de l'attribut déterminé est stockée.
- Pourquoi devrait-elle être stockée?  
*Lorsque la fonction n'est pas connue (ou son calcul est impraticable) et que toutes les valeurs en sont connues.*

## Définitions

### Discussion (2)

- Ce qui nous emmène à définir plus rigoureusement la dépendance fonctionnelle:
  - Un attribut dépend fonctionnellement d'un ensemble d'autres attributs si et seulement si (la valeur de) ces derniers permettent toujours de déterminer la valeur (unique) du premier.
- La redondance survient quand plusieurs tuples ont les mêmes valeurs pour les attributs déterminants (l'attribut déterminé apparaît lui aussi en ces mêmes tuples «de façon redondante»).

## Définitions

### Exemple 1

- Dans une université, étant donné le matricule d'un étudiant, on peut déterminer son nom et ce nom est unique.
- Il existe donc une dépendance fonctionnelle entre matricule et nom  
$$\text{matricule} \rightarrow \text{nom}$$
- L'inverse n'est pas vrai: étant donné un nom, on ne peut (en général) déterminer le matricule d'un étudiant, car il peut y avoir plusieurs étudiants de même nom, chaque étudiant ayant un matricule distinct.



## Définitions

### Précisions

- La DF [matriculeE  $\rightarrow$  nom] ne signifie pas que le nom associé à un matricule ne change jamais; le nom peut changer, mais, en tout temps, on peut déterminer le nom d'un étudiant à partir de son matricule.
- Cela ne signifie pas non plus qu'à deux matricules différents, les noms associés doivent être différents (bien qu'ils puissent l'être).
- Cela signifie qu'à un instant donné, une variable de relation (relvar) n'associe qu'un et un seul nom à un matricule.

## Définitions

### Irréductibilité et trivialité (1)

#### ○ Définitions

- Une DF  $[A \rightarrow B]$  est **triviale**  
si et seulement si  $B$  est un sous-ensemble de  $A$ .
- Une DF  $[A \rightarrow B]$  est **irréductible**  
si et seulement si aucun sous-ensemble propre de  $A$  ne détermine  $B$ .
- Une DF  $[A \rightarrow B]$  est **applicable** à une relation  $R$   
si et seulement si  $A$  et  $B$  sont sous-ensembles de l'entête de  $R$ .

#### ○ Théorème

- Les dépendances triviales sont toujours satisfaites.

## Définitions

### Irréductibilité et trivialité (2)

#### ○ Synonymes

- Une dépendance irréductible est aussi dite complète.

## Définitions

### Clé candidate

- S'il existe une DF irréductible entre  $\{A_1, \dots, A_k\}$  et tous les autres attributs  $A_{k+1}, \dots, A_n$  d'une relation  $R$ , alors  $\{A_1, \dots, A_k\}$  est une clé candidate de  $R$ .
- Réciproquement, toute clé candidate  $\{A_1, \dots, A_k\}$  de  $R$  définit une DF irréductible vers chacun des autres attributs  $A_{k+1}, \dots, A_n$  de  $R$ .
- *Notation*
  - Ces DF sont désignées les DF «directement induites» par (les clés de)  $R$ .

## Définitions

### Règles d'inférence d'Armstrong

1. réflexivité : si  $X \supseteq Y$ , alors  $X \rightarrow Y$
2. augmentation : si  $X \rightarrow Y$ , alors  $XZ \rightarrow YZ$
3. transitivité : si  $X \rightarrow Y$  et  $Y \rightarrow Z$ , alors  $X \rightarrow Z$
4. décomposition : si  $X \rightarrow YZ$ , alors  $X \rightarrow Y$  et  $X \rightarrow Z$
5. union : si  $X \rightarrow Y$  et  $X \rightarrow Z$ , alors  $X \rightarrow YZ$
6. pseudo-transitivité : si  $X \rightarrow Y$  et  $WY \rightarrow Z$ , alors  $WX \rightarrow Z$

## Définitions

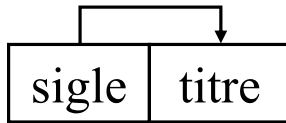
### Comment les découvrir?

- Les DF sont des propriétés issues du domaine d'application.
- On les détermine à partir de notre connaissance des faits (règles, conditions, etc.) du domaine d'application.
- On peut déceler la présence une dépendance fonctionnelle  $\{A_1, \dots, A_n\} \rightarrow A_{n+1}$  en répondant à la question suivante:
  - Lorsque toutes les valeurs des attributs  $A_1, \dots, A_n$  sont connues, peut-on **toujours** associer une et une seule valeur à l'attribut  $A_{n+1}$ ?

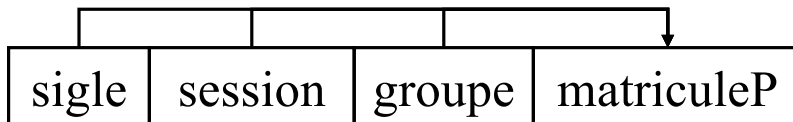
## Définitions

### Représentation graphique

○ sigle  $\rightarrow$  titre



○ sigle, session, groupe  $\rightarrow$  matriculeP  
(sigle, session, groupe)  $\rightarrow$  matriculeP



# Forme normale de Boyce-Codd (FNBC)

- Définition
- Critère



## Dépendances fonctionnelles

### Définition de la FNBC

- Une relation  $R$  est en FNBC relativement à une DF applicable non triviale  $[X \rightarrow A]$  si et seulement si
  - $X$  est une sur-clé de  $R$ .

### Rappel

- Un ensemble d'attributs est une **sur-clé** relativement à une relation  $R$ , s'il contient une clé candidate de  $R$ .

### Notation

- *On dit aussi que la DF  $[X \rightarrow A]$  doit être «induite» d'une clé (de  $R$ ).*
- *Une relation  $R$  est dite en FNBC ssi toutes les DF non triviales applicables (à  $R$ ) sont issues d'une des clés (de  $R$ ).*

## Dépendances fonctionnelles

### Critère FNBC

- Pour démontrer qu'une relation  $R$  est en FNBC, il suffit de montrer que chacune des DF non triviales qui lui sont applicables est dérivée (grâce aux axiomes d'Armstrong) des DF directement induites par les clés de  $R$ .

## Dépendances fonctionnelles

### Contre-exemple de FNBC

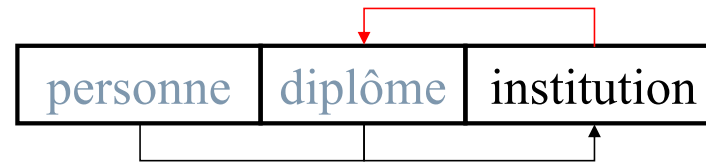
Supposons

1. qu'une institution d'enseignement décerne un seul type de diplôme (DES, DEC, BAC);
2. qu'une personne obtient un diplôme d'une et une seule institution.

On a alors les DF suivantes:

1.  $[\text{institution} \rightarrow \text{diplôme}]$
2.  $[\text{personne}, \text{diplôme} \rightarrow \text{institution}]$

Une relation R est en FNBC si et seulement si, pour toute DF non triviale  $[X \rightarrow A]$  applicable à R, X est une sur-clé.



**Cette relation n'est pas en FNBC**  
relativement à la

DF  $[\text{institution} \rightarrow \text{diplôme}]$   
car l'institution n'est pas une sur-clé.

## Dépendances fonctionnelles

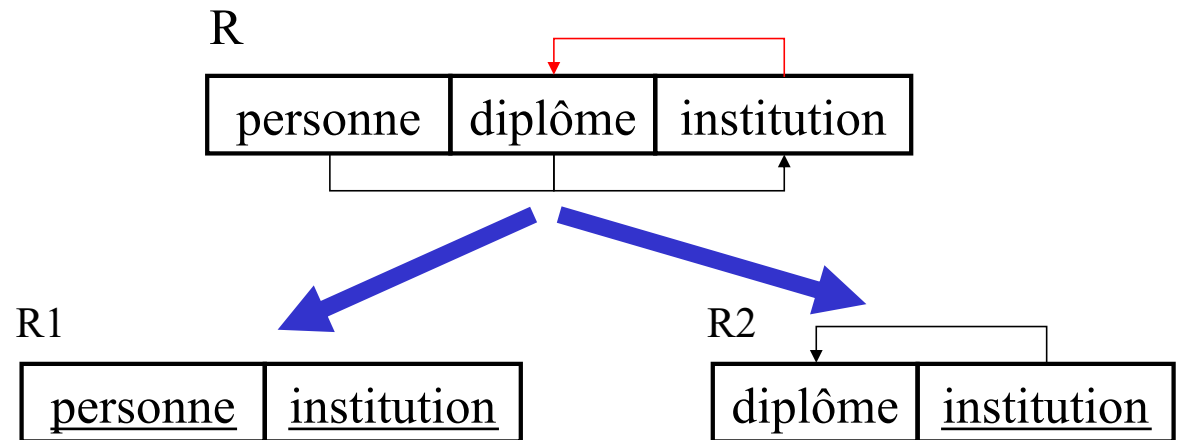
### Normalisation en FNBC

#### Remarque

- on ne perd pas d'information,
- on diminue la redondance,
- mais on perd la DF

[personne, diplôme → institution]

puisqu'elle n'est plus applicable!



Il faut donc ajouter une contrainte qui «simule» la clé candidate {personne, diplôme} sur la jointure des deux nouvelles relvar R1 et R2:

with ( $R := R1 \bowtie R2$ ) :

$$\#R = \#(R \pi \{personne, diplôme\})$$

## Dépendances fonctionnelles

### Traduction en SQL

#### ○ Traduire cette contrainte en SQL

- avec un invariant :

```
create assertion A as check(  
  with  
    R as (select * from R1 natural join R2),  
    S as (select distinct personne, diplome from R)  
  select (select count(*) from R) = (select count(*) from S)  
);
```

- avec un déclencheur et un automatisme (TRIGGER):
  - laissé en exercice

## Dépendances fonctionnelles

### Théorème de Heath

#### ○ Théorème

- Soit  $R$ , une relation dont l'entête est  $E$ ,
- soit  $X$ ,  $Y$  et  $Z$  des sous-ensembles de  $E$  tels que leur union est égale à  $E$ ,
- si  $R$  est conforme à la DF:  $X \rightarrow Y$  alors  $R$  est égal à la jointure de ses projections sur  $X \cup Y$  et  $X \cup Z$ .

#### ○ Corolaire

- La FNBC est la forme normale ultime relativement aux DF.

#### ○ Exercice

- *Pourquoi n'est-il pas nécessaire d'exiger que  $X$ ,  $Y$  et  $Z$  soient disjoints?*

# Autres formes normales relatives aux DF

- 2FN
- 3FN
- Comparaison 2FN-3FN-FNBC

*2FN*



## Dépendances fonctionnelles

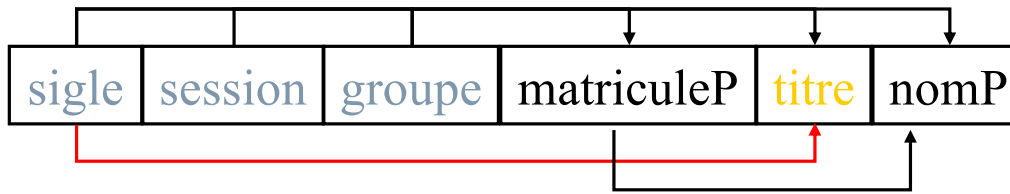
### Définition de la 2FN

- Une relation  $R$  est en 2FN si et seulement si tous les attributs **non premiers** de  $R$  sont en *dépendance fonctionnelle irréductible* de chaque clé candidate de  $E$ .
- Rappels
  - Un attribut est **premier** si et seulement s'il est élément d'une clé candidate.
  - Une *dépendance fonctionnelle est irréductible* si et seulement si aucun attribut ne peut être retiré du déterminant sans invalider la dépendance.

## Dépendances fonctionnelles

### Contre-exemple de 2FN

Une entité E est en deuxième forme normale si et seulement si tous les attributs *non premiers* de E sont en *dépendance fonctionnelle complète* de chaque *clé candidate* de E



titre ne dépend pas de toute la clé; il dépend seulement de sigle.

## Dépendances fonctionnelles

### Définition de la 2FN (bis)

- Une relation  $R$  est en 2FN si et seulement si, pour chaque DF applicable non triviale  $[X \rightarrow A]$ , une des conditions suivantes est satisfaite:
  - $X$  est une sur-clé,
  - $A$  est premier,
  - $X$  n'est pas une sous-clé.
- Rappels
  - Un attribut est **premier** si et seulement s'il est élément d'une clé candidate.
  - Un ensemble d'attributs est une **sous-clé** relativement à une relation  $R$ , s'il est un sous-ensemble d'une clé candidate de  $R$ .
  - Un ensemble d'attributs est une **sur-clé** relativement à une relation  $R$ , s'il contient une clé candidate de  $R$ .

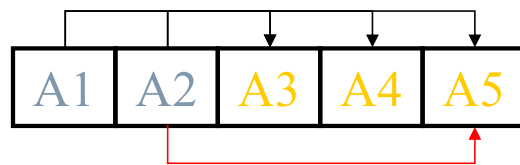
## Dépendances fonctionnelles

### Normalisation 2FN

- Les attributs non premiers en dépendance partielle sont extraits
  - pour former une nouvelle relation avec les attributs de l'une de leurs clés irréductibles présentes dans la relation,
  - ou bien,
  - sont ajoutés à une relation existante ayant une clé candidate appropriée.

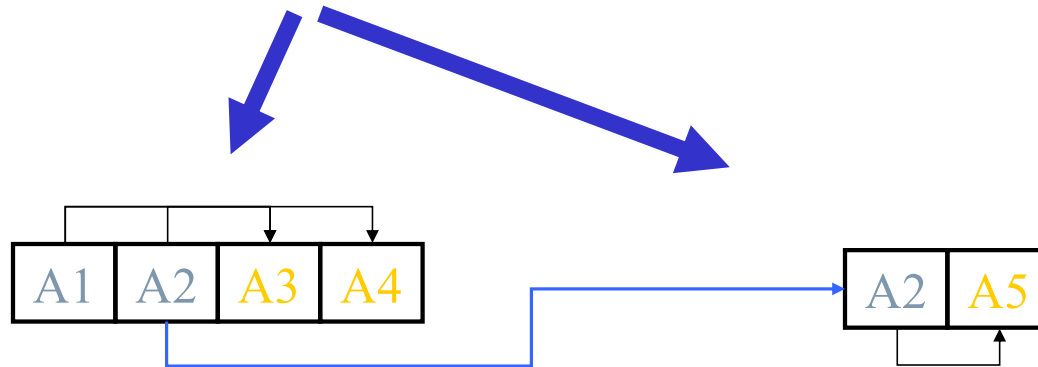
## Dépendances fonctionnelles

### Exemples de normalisation en 2FN



La relation n'est pas en 2FN, car

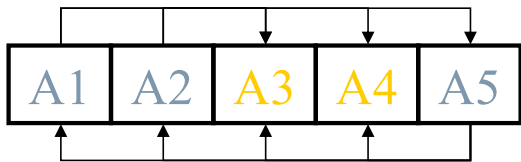
- la clé candidate est {A1,A2}
- A5 est non premier
- A5 dépend seulement de {A2}



L'ajout d'une **clé référentielle** complète la garantie d'intégrité.  
Est-ce obligatoire? Pourquoi?

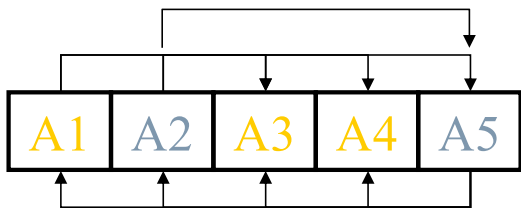
## Dépendances fonctionnelles

### Sont-elles en 2FN?



Oui

- il y a deux clés candidates  $\{A1, A2\}$  et  $\{A5\}$
- seuls les attributs A3 et A4 ne sont pas premiers
- A3 et A4 dépendent complètement de toutes les clés candidates



Oui

- il y a deux clés candidates  $\{A2\}$  et  $\{A5\}$
- seuls les attributs A1, A3 et A4 ne sont pas premiers
- A1, A3 et A4 dépendent complètement de toutes les clés candidates

*3FN*

## Dépendances fonctionnelles

### Définition de 3FN

- Une relation  $R$  est en 3FN si et seulement si, pour chaque DF applicable non triviale  $[X \rightarrow A]$ , une des conditions suivantes est satisfaite:
  - $X$  est une sur-clé,
  - $A$  est premier.
- *Une relation 2FN est donc promue en 3FN par l'élimination de la troisième condition de la 2FN:*
  - ~~$X$  n'est pas une sous-clé.~~



## Dépendances fonctionnelles

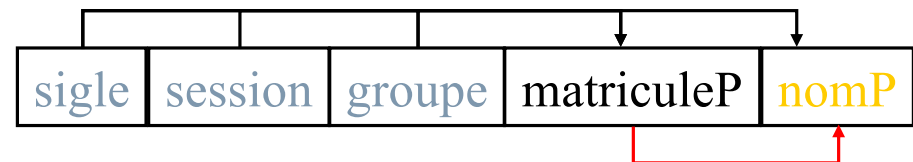
### Contre-exemple de 3FN

Une relation R est en 3FN si et seulement si, pour chaque DF non triviale  $[X \rightarrow A]$  applicable, une des conditions suivantes est satisfaite:

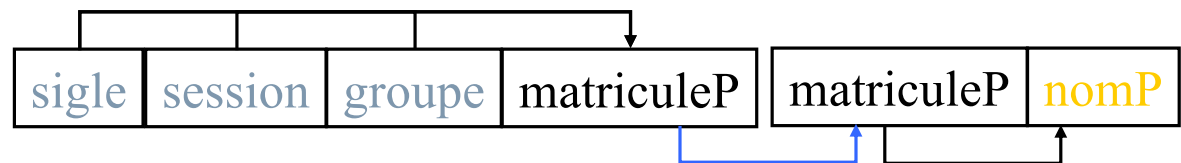
- X est une **sur-clé**,
- A est un **attribut premier**.

La relation suivante n'est pas 3FN, car

- matricule n'est pas une sur-clé,
- nom n'est pas premier.



Partition et normalisation 3FN

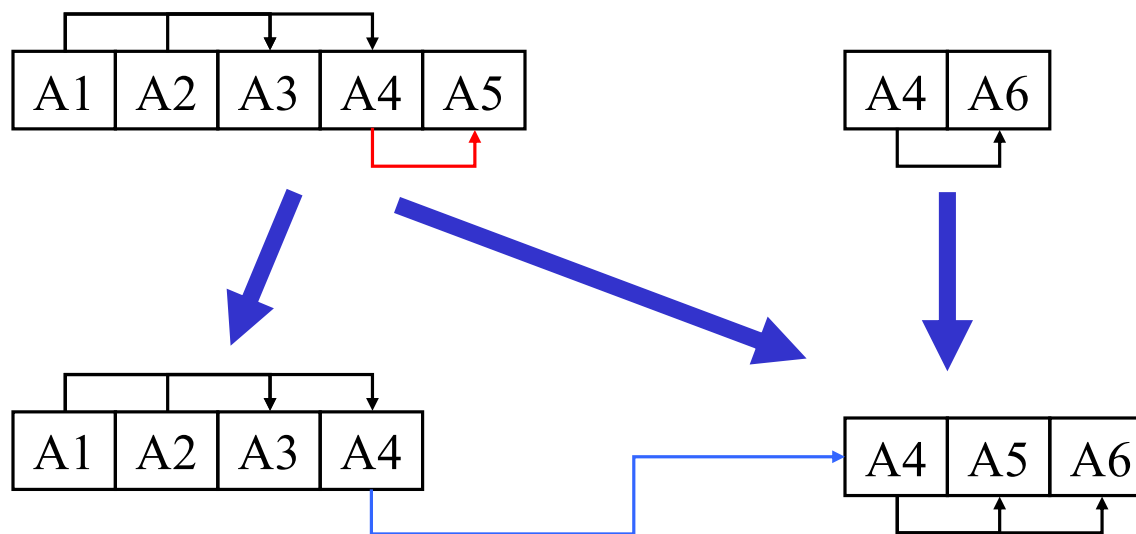


L'ajout d'une **clé référentielle** complète la garantie d'intégrité.

Est-ce obligatoire? Pourquoi?

## Dépendances fonctionnelles

### Normalisation en 3FN + Fusion

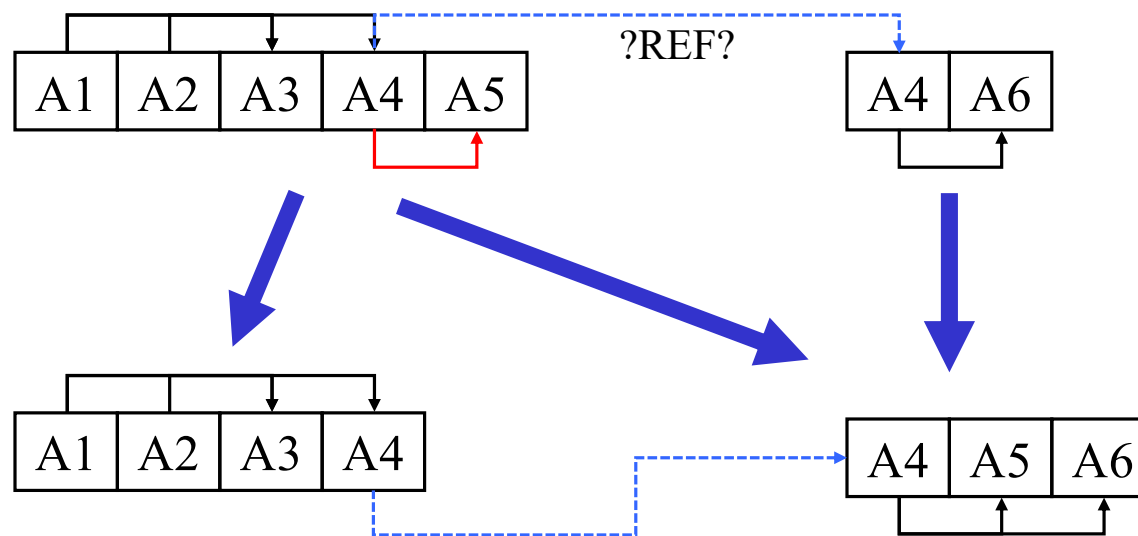


Toutes choses étant égales par ailleurs, il est avantageux de réduire le nombre de relations, d'où la fusion de  $\{A4, A5\}$  et  $\{A4, A6\}$  en  $\{A4, A5, A6\}$ .

*Est-ce toujours approprié? Pourquoi?*

## Dépendances fonctionnelles

### Normalisation en 3FN + Fusion (bis)



Faut-il ajouter une clé référentielle sur A4? Pourquoi?  
Que faut-il penser de la fusion dans ce contexte?

# *Comparaison 2FN, 3FN, FNBC*

## Dépendances fonctionnelles

### Définition de la FNBC (bis)

- Une relation  $R$  est en FNBC relativement une DF non triviale  $[X \rightarrow A]$  applicable si et seulement si
  - $X$  est une sur-clé de  $R$ .
- Une relation 3FN est donc promue en FNBC par l'élimination de la deuxième condition de la 3FN:
  - ~~$A$  est premier.~~

## Dépendances fonctionnelles

### Comparaison

- Soit les conditions suivantes relativement à une DF non triviale  $[X \rightarrow A]$  applicable à R:
  1. X est une sur-clé,
  2. A est premier,
  3. X n'est pas une sous-clé.
- Si *chacune* des DF non triviales  $[X \rightarrow A]$  applicables à R répond
  - à la condition 1, alors R est en **FNBC**,
  - à l'une des conditions 1 ou 2, alors R est en **3FN**,
  - à l'une des conditions 1, 2 ou 3, alors R est en **2FN**.
- **Conclusion**
  - *Choisissez la simplicité, choisissez FNBC!*

## Dépendances fonctionnelles

### Discussion (1)

- La 3FN peut toujours être atteinte en préservant toutes les DF sans perte de complétude, ni perte d'efficacité relativement à la 2FN.
- La 2FN est donc inintéressante puisqu'elle autorise des redondances que la 3FN permet d'éliminer.
- La FNBC ne peut pas toujours être atteinte sans perte de DF, il serait donc tentant de se rabattre parfois sur la 3FN.
- Par contre, la DF «perdues» peuvent toujours être mises en oeuvre à l'aide de contraintes (ASSERTION ou TRIGGER, en SQL).

## Dépendances fonctionnelles

### Discussion (2)

- Le choix entre la 3FN et la FNBC repose sur l'évaluation des éléments suivants
  - le développement des contraintes supplémentaires (ressources, délais, etc.) ;
  - la perte (éventuelle) d'efficacité de la BD en raison de l'ajout de ces contraintes ;
  - l'impact de l'absence de ces contraintes sur la cohérence, la validité et l'efficacité de la BD.
- Autrement dit
  - Est-il préférable d'arriver (un peu) plus rapidement à une réponse (possiblement) invalide ou d'arriver (un peu) plus lentement à une réponse valide ?



# Dépendances de jointure

- Exemple A — une clé
- Définition de la 5FN — cas simple
- Exemple A — une clé (suite)
- Induction
- Définition de la 5FN — cas général
- Retour sur le cas simple
- Exemple B — deux clés
- Théorème général de P-J

## Dépendances de jointure

### Exemple A (1)

- Soit la relvar

OffreDeCours {sigle, session, matriculeP}

dont le prédicat est le suivant :

Le professeur «matriculeP» donne le cours «sigle»  
à la session «session».

- La clé de la relvar OffreDeCours est  
{sigle, session, matriculeP}.

## Dépendances de jointure

### Exemple A (2)

OffreDeCours peut-elle représenter également les trois relations suivantes?

- Disponibilité {matriculeP, session}
  - le professeur «matriculeP» est disponible pour enseigner à la session «session»;
- Compétence {matriculeP, sigle}
  - le professeur «matriculeP» est apte à enseigner l'activité «sigle»;
- Offre {sigle, session}
  - l'activité «sigle» est programmée à la session «session».

## Dépendances de jointure

### Exemple A (3)

Avant de répondre à la question précédente, envisager les prédicats suivants :

- Disponibilité {matriculeP, session, charge}
  - le professeur «matriculeP» est disponible pour enseigner à la session «session» au plus «charge» cours;
- Compétence {matriculeP, sigle, delai}
  - le professeur «matriculeP» est apte à enseigner l'activité «sigle» moyennant un préavis de « delai » semaine;
- Offre {sigle, session, priorité}
  - l'activité «sigle» est programmée à la session «session» avec la priorité «priorité».

## Dépendances de jointure

### Exemple A (4)

- Manifestement, la réponse est non, la relation  
OffreDeCours {sigle, session, matriculeP}  
ne comporte pas les attributs charge, délai ni priorité.
- Qu'en serait-il alors de la relation  
OffreDeCours {sigle, session, matriculeP, horaire, charge, délai,  
priorité}  
dont les DF seraient les suivantes
  - sigle, session, matriculeP -> horaire
  - session, matriculeP -> charge
  - sigle, matriculeP -> délai
  - sigle, session -> priorité
- Elle ne serait manifestement pas en FNBC.
- Il faudrait définir quatre relations, une par dépendance.

## Dépendances de jointure

### Exemple A (5)

Enlever les attributs non clés rend-elle la situation plus acceptable ?

Revenons à notre questions initiale : OffreDeCours peut-elle représenter également les trois relations suivantes?

- Disponibilité {matriculeP, session}
  - le professeur «matriculeP» est disponible pour enseigner à la session «session»;
- Compétence {matriculeP, sigle}
  - le professeur «matriculeP» est apte à enseigner l'activité «sigle»;
- Offre {sigle, session}
  - l'activité «sigle» est programmée à la session «session».

## Dépendances de jointure

### Exemple A (6)

Réponse :

Non, car OffreDeCours n'est pas la jointure des trois autres relations.

Par exemple:

OffreDeCours		
<u>sigle</u>	<u>session</u>	<u>matriculeP</u>
IFT 333	2014-1	1
IFT 333	2014-3	2

≠

Offre	
<u>sigle</u>	<u>session</u>
IFT 333	2014-1
IFT 333	2014-3



Disponibilité	
<u>matriculeP</u>	<u>session</u>
1	2014-1
2	2014-1
2	2014-3



Compétence	
<u>matriculeP</u>	<u>sigle</u>
1	IFT 333
2	IFT 333

## Dépendances de jointure

### Exemple A (7)

- Les contrainte qui lient les quatre tables sont plutôt les clés référentielles d'Offre\_de\_cours vers chacune des autres :
  - ref {sigle,session} -> Offre
  - ref {matriculeP,session} -> Disponibilité
  - ref {matriculeP,sigle} -> Compétence



## Dépendances de jointure

### Définition

- $\bowtie\{d_1, \dots, d_k\}$  est une dépendance de jointure (DJ) sur  $E$ , si et seulement si
  - $d_1 \subseteq E, \dots, d_k \subseteq E$
  - $E = d_1 \cup \dots \cup d_k$
- Un relation  $R$  satisfait une DJ si et seulement si
  - $R = (R \pi d_1) \bowtie \dots \bowtie (R \pi d_k)$

### Notation

- Chaque  $d_i$  est appelé composant(e) de la DJ.

## Dépendances de jointure

### Exemple A (8)

- Dans notre exemple, la dépendance de jointure à satisfaire est
  - $\bowtie \{ \{ \text{session}, \text{matriculeP} \}, \{ \text{sigle}, \text{matriculeP} \}, \{ \text{sigle}, \text{session} \} \}$
- Or, elle ne peut être satisfaite en général (voir l'exemple).

## *5FN – cas simple*

## Dépendances de jointure

### 5FN – cas simple

- Une relation  $R$  ayant une seule clé candidate est en cinquième forme normale (5FN) relativement à une DJ si tous ses composants sont des surclés de  $R$ .
- Dans ce cas, on démontre que
  - $R = (R \pi d_1) \bowtie \dots \bowtie (R \pi d_k)$
- Une relation est en 5FN si elle l'est relativement à chacune des DJ qui lui sont applicables.

## Dépendances de jointure

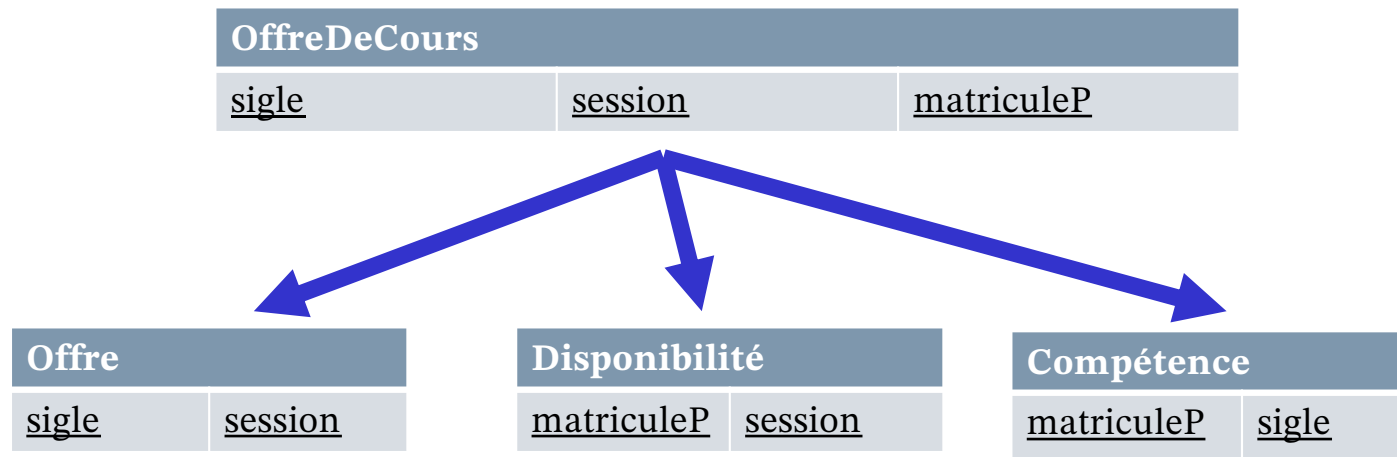
### Exemple A — retour (1)

- La relvar OffreDeCours n'est pas en 5FN, car ses composants ne sont pas des sur-clés. En effet, la (seule) clé de OffreDecours est {sigle, matriculeP, session}.
- Corolaire: OffreDeCours représente un prédicat différent de la jointure des composantes. Et c'est ce que nous avons montré.
- Question
  - Comment changer la modélisation?
  - Pourquoi est-ce important?

## Dépendances de jointure

### Exemple A — retour (2)

- Il faut donc les maintenir indépendamment et ajouter 3 contraintes référentielles.



## Dépendances de jointure

### Exemple A — retour (3)

Les prédicats sont les suivants

- Disponibilité {matriculeP, session}
  - le professeur *peut enseigner* durant cette session.
- Compétence {matriculeP, sigle}
  - le professeur *est apte* à enseigner le cours identifié par sigle.
- Offre {sigle, session}
  - le cours identifié par sigle *est susceptible d'être offert* à cette session.
- OffreDeCours {matriculeP, session, sigle}
  - le professeur *enseigne* le cours identifié par sigle durant cette session.

## Dépendances de jointure

### Exemple A — retour (4)

- Deux propositions peuvent être avancées:
  - Il y a de la redondance, mais elle est contrôlée par les clés référentielles.
  - Il n'y a pas de redondance, car ce sont des prédicats différents.
- Par contre, une chose est certaine: le programmeur devra aller chercher la bonne information au bon endroit (donc, connaître les prédicats).
- Que fera-t-il s'il ne connaît que l'entête des relations, mais pas les prédicats?
- Comment qualifieriez-vous un concepteur de BD qui ne documenterait pas toutes ses relvars?



## *5FN – cas général*

- Enfin !

## Dépendances de jointure

### Induction

- Une DJ est induite par une clé de R si et seulement si la toutes les composantes de DJ comprennent cette clé.

## Dépendances de jointure

### 5FN – cas général

Une relation  $R$  est en cinquième forme normale (5FN) si et seulement si chacune des DJ qui lui sont applicables est induite par une des clés de  $R$ .

Dans ce cas, on démontre que

$$R = (R \pi d_1) \bowtie \dots \bowtie (R \pi d_k)$$

en regard de toutes les DJ qui lui sont applicables.

## Dépendances de jointure

### Retour sur le cas simple

- Lorsque R ne comprend qu'une clé candidate:
  - Une DJ est induite par la clé de R si et seulement si toutes les composantes comprennent la clé (sont une surclé) de R.

## Dépendances de jointure

### Exemple B — deux clés

R relvar

{cip, admAn, admNo, nom, prenom, sexe, programme},  
key {cip}, {admAn, admNo} ;

DJ { {cip, nom, prenom},  
{cip, sexe},  
{cip, programme},  
{cip, admAn, admNo} };

DJ { {admAn, admNo, cip},  
{admAn, admNo, nom, prenom},  
{admAn, admNo, sexe},  
{admAn, admNo, programme} };

## Dépendances de jointure

### Exemple B — deux clés (version «normalisée»)

R relvar

{cip, admAn, admNo, nom, prenom, sexe, programme},  
key {cip}, {admAn, admNo} ;

DJ {    {cip, nom},  
         {cip, prenom},  
         {cip, sexe},  
         {cip, programme},  
         {cip, admAn},  
         {cip, admNo} };

DJ {    {admAn, admNo, cip},  
         {admAn, admNo, nom},  
         {admAn, admNo, prenom},  
         {admAn, admNo, sexe},  
         {admAn, admNo, programme} };

## Dépendances de jointure

### Théorème général de projection-jointure

- Soit  $R$  une relvar dont l'entête est  $E$ .
- Soit les composants  $d_1, \dots, d_k$  des sous-ensembles de  $E$  tels que leur union est égale à  $E$ .
- Alors  $R$  est égale à la jointure de ses projections sur les composants  $d_i$  si et seulement si  $R$  est en 5FN relativement à  $\bowtie\{d_1, \dots, d_k\}$ .

# Dépendances multivaluées

- Exemple 1
- Définition
- 4FN
- Analyse de l'exemple 1
- Exemple 2
- Normalisation
- Théorème de Fagin

**La dépendance multivaluée est un cas particulier de dépendance de jointure admettant une interprétation spécifique plus facilement identifiable lors de l'analyse de modélisation.**



## Dépendances multivaluées

### Exemple 1

*On veut s'assurer que tous les enseignants utilisent les mêmes livres pour un même cours.*

*Ce n'est pas le cas ici, Zoroastre est délinquant!*

<u>cours</u>	<u>enseignant</u>	<u>livre</u>
IFT 187	Marc	Elmasri
IFT 187	Marc	Frappier
IFT 187	Luc	Elmasri
IFT 187	Luc	Frappier
IFT 487	Luc	Elmasri
IFT 487	Luc	Date
IFT 187	Zoroastre	Elmasri
IFT 187	Zoroastre	Frappier
IFT 187	Zoroastre	Date

*(Les tuples associés à) l'enseignant Zoroastre ne respecte(nt) pas la 4FN.*

## Dépendances multivaluées

### Définition

- Soit  $X$  et  $Y$  deux sous-ensembles d'attributs d'une relation  $R$ , une DM  $X \twoheadrightarrow Y$  représente le fait que «tout tuple ayant les mêmes valeurs en  $X$  est associé à un seul et même *ensemble de valeurs* en  $Y$ ».
- Rappel
  - une DF  $X \rightarrow Y$  peut être interprétée comme «tout tuple ayant la même valeur en  $X$  est associé à une seule et même *valeur* en  $Y$ ».

## Dépendances multivaluées

### 4FN

Soit

- E l'entête de R et
- $Z = E - (X \cup Y)$

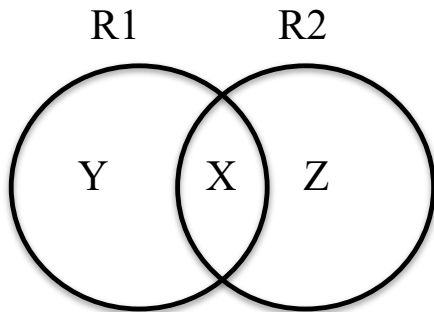
R est en 4FN relativement à la DM  $X \twoheadrightarrow Y$  si et seulement si

- $(X \cup Y)$  est une surclé de R
- On démontre alors que
  - $(R \pi (X \cup Y)) \bowtie (R \pi (X \cup Z)) = R$
- Remarquer la symétrie de la jointure et conclure que
  - $X \twoheadrightarrow Y$  induit nécessairement  $X \twoheadrightarrow Z$  et réciproquement.
- Corolaire
  - Y et Z sont indépendants
  - bien que *chacun* soit multidéterminé par X.

## Dépendances multivaluées

Un cas particulier de dépendances de jointures!

- Soit la relation R dont l'entête  $E = R1 \cup R2$



- Équivalences
  - $X \twoheadrightarrow Y$
  - $X \twoheadrightarrow Z$
  - $\bowtie \{X \cup Y, X \cup Z\}$
  - $\bowtie \{R1, R2\}$
  - $R1 \cap R2 \twoheadrightarrow R1 - R2$
  - $R1 \cap R2 \twoheadrightarrow R2 - R1$

## Dépendances multivaluées

### Exemple 1

#### Analyse

- La relation  $S(c, e, x)$  représente le prédicat «l'enseignant  $e$  donne le cours  $c$  à l'aide du livre  $x$ ».
- Les règles suivantes sont applicables:
  - un enseignant peut donner plus d'un cours;
  - un cours peut être donné à l'aide de plusieurs livres;
  - pour un cours donné, tous les enseignants utilisent les mêmes livres.
- Ces règles peuvent être résumées par la DM  $\{c\} \twoheadrightarrow \{e\}$  (ce qui induit implicitement la DM  $\{c\} \twoheadrightarrow \{x\}$ ).
- On remarque que la DF  $\{c\} \rightarrow \{e\}$  n'est pas applicable, puisqu'un cours peut déterminer plus d'un enseignant.
- Le même raisonnement s'applique à DF  $\{c\} \rightarrow \{x\}$ .
- Par contre,
  - quel que soit le livre, tous les enseignants donnant le cours l'utiliseront;
  - quel que soit l'enseignant, tous les livres utilisés par le cours seront utilisés par l'enseignant.
- Corolaire: les enseignants sont indépendants des livres, bien que chacun soit déterminé par le cours.

## Dépendances multivaluées

### Exemple 2

<u>nom</u>	<u>adresse</u>	<u>ville</u>	<u>emploi</u>
Pauline	12, rue de la gare	Saint-Jean	géomaticienne
Pauline	1250, rue X	Montréal	géomaticienne
Luc	50, rue du domaine	Saint-Donat	informaticien
Luc	10, rue des seigneurs	Sherbrooke	informaticien
Luc	50, rue du domaine	Saint-Donat	enseignant
Luc	10, rue des seigneurs	Sherbrooke	enseignant
Claude	1, rue principale	Saint-Donat	écologiste
Claude	1, rue principale	Saint-Jean	botaniste
Claude	1, rue principale	Saint-Donat	botaniste
Claude	1, rue principale	Saint-Jean	écologiste
Zoroastre	100, rue Y	Montréal	peintre
Zoroastre	200, rue Z	Montréal	actuaire

***Zoroastre ne respecte pas la 4FN***

## Dépendances multivaluées

### Exemple 2

#### Analyse

- La relation  $S(n, a, v, e)$  représente le prédicat «la personne portant le nom  $n$  habite à l'adresse  $a$  de la ville  $v$  et occupe l'emploi  $e$ ».
- Les règles suivantes sont applicables
  - une personne peut habiter à plusieurs endroits (un endroit est déterminé par le couple adresse, ville);
  - une personne peut occuper plusieurs emplois;
  - les emplois et les lieux d'habitation sont indépendants.
- Ces règles peuvent être résumées par la DM  $\{n\} \twoheadrightarrow \{a, v\}$  (ou de façon équivalente  $\{n\} \twoheadrightarrow \{e\}$ ).
- On remarque que la DF  $\{n\} \rightarrow \{a, v\}$  n'est pas applicable, puisqu'une personne peut déterminer plus d'un endroit (adresse, ville).
- Le même raisonnement s'applique à la DF  $\{n\} \rightarrow \{e\}$ .
- Par contre, quel que soit l'endroit, toute personne a les mêmes emplois (quel que soit l'emploi, la personne habite les mêmes endroits).
- Corolaire: les emplois sont indépendants des endroits.

## Dépendances multivaluées

### Normalisation

- Scinder la relation en deux
- Ajouter les clés référentielles
- Voir la normalisation 5FN!



## Dépendances multivaluées

### Théorème de Fagin

- Soit  $R$  une relation dont l'entête est  $E$ ,
- Soit  $X$ ,  $Y$  et  $Z$  des sous-ensembles de  $E$  tels que leur union est égale à  $E$ ,
- $R$  est égal à la jointure de ses projections sur  $X \cup Y$  et  $X \cup Z$  si et seulement si  $X \twoheadrightarrow Y$ .
- La 4FN est donc l'ultime forme normale pour les dépendances multivaluées.
- Rappel  $(X \twoheadrightarrow Y) \Leftrightarrow (X \twoheadrightarrow Z)$

# Quelques théorèmes utiles

- $\text{FNBC} + \text{ANC} \Rightarrow 5\text{FN}$
- $3\text{FN} + 0\text{CCC} \Rightarrow 5\text{FN}$
- $\text{RT} + 0\text{DFNT} \Leftrightarrow \text{RT} + \text{FNBC}$
- Synthèse
- Pour aller un peu plus loin... la 6FN

## Quelques théorèmes utiles

FNBC + ANC  $\Rightarrow$  5FN

- Toute relation FNBC comportant au moins un attribut non-clé est en 5FN.

## Quelques théorèmes utiles

3FN + 0CCC  $\Rightarrow$  5FN

- Toute relation 3FN (*a fortiori* FNBC) ne possédant aucune clé candidate composite est en 5FN.

## Quelques théorèmes utiles

$RT + 0DFNT \Leftrightarrow RT + FNBC$

- Une relation totale n'est en FNBC que si et seulement si le déterminant de chacune des DF applicables est l'entête de la relation!
- En définitive, une relation totale n'est en FNBC que s'il n'y a aucune DF non triviale!
- RAPPEL
  - Une relation totale qui n'est pas en FNBC ne peut être en 5FN !

## Quelques théorèmes utiles

### Rappel : relations totales

- Si l'entête d'une relation est une clé candidate de celle-ci, elle en est la seule.
- *Une telle relation est désignée comme étant une relation «totale» (RT).*
- Une relation totale est en 3FN (par définition).

## Quelques théorèmes utiles

### Synthèse

- Les formes «ultimes»
  - DF: FNBC (théorème de Heath)
  - DM: 4FN (théorème de Fagin)
  - DJ: 5FN (théorème général de projection-jointure)

## Pour aller un peu plus loin... la 6FN

- Une relvar est en sixième forme normale (6FN) si et seulement si, quelle que soit la dépendance de jointure à laquelle elle satisfait, cette dépendance est triviale.
- Une relvar est en 6FN si et seulement si elle est en 5FN et n'a pas plus d'un attribut non clé.
- Une relvar est en 6FN si et seulement si elle ne peut être décomposée par projection-jointure en relations de degré inférieur.



# Les algorithmes

- Noyau
  - couverture irréductible
- FNBC
  - avec préservation des jointures
- 3FN
  - avec préservation des dépendances
- Références

## Calcul du noyau — esquisse

- Un noyau d'un ensemble de dépendances est ensemble minimal de dépendances normalisées permettant de toutes les dériver.
- Une dépendance est normalisée si elle est de forme  $[X \rightarrow A]$ .
- Soit  $F$  un ensemble de dépendances  $\{D_1, \dots, D_n\}$ 
  - Soit  $E$ , l'ensemble normalisé de  $F$ .
  - Examiner chaque  $D_i$  et retirer tous les attributs du déterminant dont le retrait laisse  $E^+$  invariant.
  - Retirer de  $E$  toute dépendance laissant  $E^+$  invariant.
  - L'ensemble  $E$  résultant est un noyau de  $F$ .

## Calcul du noyau — Elmasri, p. 550

**Definition.** A minimal cover of a set of functional dependencies  $E$  is a minimal set of dependencies (in the standard canonical form and without redundancy) that is equivalent to  $E$ . We can always find *at least one* minimal cover  $F$  for any set of dependencies  $E$  using Algorithm 16.2.

If several sets of FDs qualify as minimal covers of  $E$  by the definition above, it is customary to use additional criteria for *minimality*. For example, we can choose the minimal set with the *smallest number of dependencies* or with the *smallest total length* (the total length of a set of dependencies is calculated by concatenating the dependencies and treating them as one long character string).

**Algorithm 16.2.** Finding a Minimal Cover  $F$  for a Set of Functional Dependencies  $E$

**Input:** A set of functional dependencies  $E$ .

1. Set  $F := E$ .
2. Replace each functional dependency  $X \rightarrow \{A_1, A_2, \dots, A_n\}$  in  $F$  by the  $n$  functional dependencies  $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_n$ .
3. For each functional dependency  $X \rightarrow A$  in  $F$   
for each attribute  $B$  that is an element of  $X$   
if  $\{F - \{X \rightarrow A\}\} \cup \{(X - \{B\}) \rightarrow A\}$  is equivalent to  $F$   
then replace  $X \rightarrow A$  with  $(X - \{B\}) \rightarrow A$  in  $F$ .
4. For each remaining functional dependency  $X \rightarrow A$  in  $F$   
if  $\{F - \{X \rightarrow A\}\}$  is equivalent to  $F$ ,  
then remove  $X \rightarrow A$  from  $F$ .

## Normalisation FNBC — esquisse (sans garantie de préservation des contraintes)

- Soit  $R$  une relation d'entête  $E$  dont  $C$  est le noyau des dépendances applicables
  - $S := E$
  - Pour chaque déterminant distinct  $X$  de  $C$ 
    - Soit  $X \rightarrow Y_1, \dots, X \rightarrow Y_n$  les dépendances qui découlent de  $X$  dans  $C$
    - Soit  $Y$  l'union des  $Y_i$
    - Ajouter l'union de  $X$  et  $Y$  à  $S$
  - Pour chaque composante  $Z$  non FNBC de  $S$ 
    - Soit  $X \rightarrow Y$  une dépendance non satisfaite
      - Remplacer  $Z$  par les composantes  $(X \cup Y)$  et  $(Z - Y)$

### 16.3.2 Nonadditive Join Decomposition into BCNF Schemas

The next algorithm decomposes a universal relation schema  $R = \{A_1, A_2, \dots, A_n\}$  into a decomposition  $D = \{R_1, R_2, \dots, R_m\}$  such that each  $R_i$  is in BCNF *and* the decomposition  $D$  has the lossless join property with respect to  $F$ . Algorithm 16.5 utilizes Property NJB and Claim 2 (preservation of nonadditivity in successive decompositions) to create a nonadditive join decomposition  $D = \{R_1, R_2, \dots, R_m\}$  of a universal relation  $R$  based on a set of functional dependencies  $F$ , such that each  $R_i$  in  $D$  is in BCNF.

**Algorithm 16.5.** Relational Decomposition into BCNF with Nonadditive Join Property

**Input:** A universal relation  $R$  and a set of functional dependencies  $F$  on the attributes of  $R$ .

1. Set  $D := \{R\}$  ;
2. While there is a relation schema  $Q$  in  $D$  that is not in BCNF do  
  {  
    choose a relation schema  $Q$  in  $D$  that is not in BCNF;  
    find a functional dependency  $X \rightarrow Y$  in  $Q$  that violates BCNF;  
    replace  $Q$  in  $D$  by two relation schemas  $(Q - Y)$  and  $(X \cup Y)$ ;  
  };

## Normalisation 3FN — esquisse (avec préservation des contraintes)

- Soit  $R$  une relation d'entête  $E$  dont  $C$  est le noyau des dépendances applicables
  - $S := \{\}$
  - Pour chaque déterminant distinct  $X$  de  $C$ 
    - Soit  $X \rightarrow Y_1, \dots, X \rightarrow Y_n$  les dépendances qui en découlent dans  $C$
    - Soit  $Y$  l'union des  $Y_i$
    - Ajouter l'union de  $X$  et  $Y$  à  $S$
  - Soit  $U$  l'ensemble des attributs  $R$  absents de  $S$ 
    - Si  $U$  est non vide, l'ajouter à  $S$
  - Si aucun élément de  $S$  n'est une surclé de  $R$ 
    - Ajouter une clé  $K$  de  $R$  à  $S$

### 16.3.1 Dependency-Preserving Decomposition into 3NF Schemas

Algorithm 16.4 creates a dependency-preserving decomposition  $D = \{R_1, R_2, \dots, R_m\}$  of a universal relation  $R$  based on a set of functional dependencies  $F$ , such that each  $R_i$  in  $D$  is in 3NF. It guarantees only the dependency-preserving property; it does *not* guarantee the nonadditive join property. The first step of Algorithm 16.4 is to find a minimal cover  $G$  for  $F$ ; Algorithm 16.2 can be used for this step. Note that multiple minimal covers may exist for a given set  $F$  (as we illustrate later in the example after Algorithm 16.4). In such cases the algorithms can potentially yield multiple alternative designs.

**Algorithm 16.4.** Relational Synthesis into 3NF with Dependency Preservation

**Input:** A universal relation  $R$  and a set of functional dependencies  $F$  on the attributes of  $R$ .

1. Find a minimal cover  $G$  for  $F$  (use Algorithm 16.2);
2. For each left-hand-side  $X$  of a functional dependency that appears in  $G$ , create a relation schema in  $D$  with attributes  $\{X \cup \{A_1\} \cup \{A_2\} \dots \cup \{A_k\}\}$ , where  $X \rightarrow A_1, X \rightarrow A_2, \dots, X \rightarrow A_k$  are the only dependencies in  $G$  with  $X$  as the left-hand-side ( $X$  is the key of this relation);
3. Place any remaining attributes (that have not been placed in any relation) in a single relation schema to ensure the attribute preservation property.

## Références

- Elmasri et Navathe, chap. 16
- Dirk Beyer, Andreas Noack, and Claus Lewerentz.  
*Efficient Relational Calculation for Software Analysis*  
IEEE Transactions on Software Engineering,  
Vol. 31, No. 2, February 2005, p.137
- François de Sainte Marie  
<https://fsmrel.developpez.com/basesrelationnelles/normalisation/>  
(consulté le 2025-03-25)





## La recette 1FN de l'ingénieur

### ○ Critère

- Tous les attributs de toutes relvars doivent être atomiques

### ○ Procédure

- Appliquer une politique de typage strict

### ○ En particulier

- Attention aux conversions implicites (souvent approximatives)
- Ne pas traiter une valeur scalaire comme une valeur non scalaire
  - Par exemple, utiliser une valeur textuelle comme «contenant» d'une liste de valeurs

### ○ Tolérance

- Définir un sous-type scalaire à l'aide d'une contrainte validant systématiquement et totalement la structure non scalaire
  - Il pourrait néanmoins en résulter une vérification moins complète, voir l'exemple des listes de cours

## La recette FNBC de l'ingénieur

### ○ Critère

- Attendu un schéma en 1FN
- Pour chaque relvar, toutes les dépendances fonctionnelles (DF) de la relvar doivent être issues de ses clés

### ○ Procédure

- Vérifier que tous les clés strictes sont dûment identifiées et déclarées
- Vérifier la clé primaire en regard de tous les attributs n'en faisant pas partie
- Vérifier que chaque clé secondaire détermine la clé primaire
- Vérifier qu'il n'existe aucune DF entre attributs non clés

## La recette 5FN de l'ingénieur

### ○ Critère

- Attendu un schéma en FNBC
- Toutes les DPJ doivent être triviales  
(seules les relations totales nécessitent une vérification)

### ○ Procédure (partielle)

- Identifier toutes les relations totales comportant une DPJ non triviale
- Pour chaque telle relvar
  - Ajouter les relvars suivantes
    - une relvar composée de toutes les clés singletons
    - une relvar pour chaque clé non singleton (relvar composée de cette seule clé singleton)
    - ajouter la contrainte vérifiant que la relvar d'origine est un sous-ensemble de la jointure des relvars ajoutées
    - adjoindre l'énoncé du prédicat appropriés à chacune des relvars ajoutées