Bases de données *Conception*

Intégrité, redondance et normalisation (version intégrale)

MLR_01a
v252a

2022-03-29

UNIVERSITÉ DE SHERBROOKE

Christina.Khnaisser@USherbrooke.ca
Luc.Lavoie@USherbrooke.ca

Plan

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

Note:
les sections en italiques peuvent être omises en première lecture.

Normalisation

- Définitions et hypothèses
- Vision traditionnelle
- Vision contemporaine
- Pourquoi réduire la redondance?
- Comment réduire la redondance?
- Le rôle des contraintes
- Pourquoi ne pas se contenter de contraintes?

- Pour être *adéquat*, un modèle logique de BD doit 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 expression relationnelle.

Hypothèses

- La redondance des données entraine indirectement la redondance des propositions, elle devient donc un facteur important d'incohérences potentielles.
- La réduction de la redondance des données réduit donc les risques d'incohérence.
- Elle facilite également l'atteinte de la validité et de l'efficacité lors de:
  - la mise à jour des données,
  - l'interrogation des données,
  - l'évolution du modèle logique.

- On distingue plusieurs formes de normalisation de modèles logiques relationnels, dont les suivantes:
  - d'attribut (1FN)
    - assure que tout attribut est associé à exactement une valeur de son type de définition;
  - de proposition (FNBC≡FNDF, FNDJ≡FNPJ≡5FN, 6FN)
    - assure que si une même proposition est représentée plus d'une fois dans la base de données, toutes ses représentations sont cohérentes;
  - d'intervalle (FNI)
    - assure l'unicité, l'intégrité et la cohérence d'une relation compacte en regard d'attributs de type intervalle;
  - historique (FNBT)
    - assure la cohérence historique prédicats bi-temporels.

Normalisation
Formes traditionnelles

- Normalisation des attributs
  - 1FN

- Normalisation des prédicats
  (donc des propositions qu'ils comprennent):
  - 2FN, 3FN, FNBC, 4FN, 5FN, 6FN, FND…

- En particulier
  - 6FN ⇒ 5FN ⇒ 4FN ⇒ FNBC ⇒ 3FN ⇒ 2FN ⇒ 1FN
  - 6FN, 5FN, FNBC et 1FN: sont les formes essentielles
  - 4FN et 3FN sont utiles à des fins d'illustration
  - 2FN est inutile en pratique

Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

Vision

- À strictement parler, seule la 1FN traite de normalisation au sens mathématique usuel, à savoir *une façon unique et uniforme de représenter correctement toute relation respectant les axiomes de la théorie relationnelle.*
- La normalisation consiste à s'assurer qu'une relation respecte les axiomes de la théorie grâce à (la transformation en) une représentation qui le garantit par sa seule structure.
- Toutes les autres «formes normales» relationnelles sont en fait des propriétés et des transformations qui visent à concevoir un modèle cohérent, complet et évolutif (et donc à réduire, voire éliminer, la redondance).

Conséquences

- 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» (autres que 1FN) sont donc des moyens d'appliquer un principe de conception dans le but de minimiser la redondance.

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 ne le corrige pas.

- La normalisation offre cependant l’opportunité de corriger certaines erreurs de conception en les rendant plus manifestes.

Une intuition

- Pour être en mesure de retirer une proposition, il est souhaitable (sinon nécessaire) qu'elle soit représentée indépendamment de toute autre proposition qui n'en découlerait pas.
- Un tuple ne devrait donc contenir qu'une proposition principale éventuellement accompagnée de propositions dépendantes.

Comment réduire la redondance? — Le principe.

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

Normalisation relative aux propositions
Comment la décomposition réduit-elle la redondance? — Un exemple.

- Parce qu'une relation est un ensemble et que les tuples multiples seront représentés par un seul!

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

Normalisation
Comment la projection réduit-elle la redondance? — Un exemple.

- Parce qu'une relation est un ensemble et que les tuples multiples seront représentés par un seul!

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

Normalisation
Persistance de redondances

- Malheureusement, la décomposition n’est pas toujours possible, ni toujours efficiente.

- Des redondances peuvent donc persister dans un modèle logique.

Comment contrôler la redondance?

- 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 entraine le retrait dans les autres emplacements.

- Pour contrôler les occurrences redondantes, on peut, *il faut*, les lier par une contrainte.

Normalisation
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⇒3FN⇒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⇒5FN).
- Plusieurs autres formes normales ont été développées, dont la 6FN qui garantit l'indépendance des prédicats élémentaires.

Normalisation en regard des attributs

1FN

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

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 v une variable de type relationnel $R\ (a_1{:}T_1, \ldots, a_n{:}T_n)$, v est normalisée en regard des attributs si et seulement si pour tout tuple t de v, 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).

Corolaires

- Un attribut ne peut être sans valeur
  (pas de marqueur NULL; l'extension de tous les types par un supremum IN et d'un infimum NA 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 ensemble de valeurs — auquel cas l'ensemble est une valeur unique).
- Toute relation **est** en 1FN *par définition*
  (voir définition donnée dans le module TRM_01 — Fondements).

Rappel TRM_01: Fondements — Tuples

- Soit a_i des identifiants distincts et D_j des types, un tuple t est défini comme suit:
  - $t \triangleq (\{a_1{:}D_1, a_2{:}D_2, \ldots, a_n{:}D_n\}; \{(a_1,v_1), (a_2,v_2), \ldots (a_n,v_n)\})$
  - avec $\forall\, i: 1 \leq i \leq deg(t) \Longrightarrow val(t, a_i) \in def(t, a_i)$
- où
  - $def(t) = \{a_1{:}D_1, a_2{:}D_2, \ldots, a_n{:}D_n\}$ entête de t
  - $def(t, a_i) = D_i$ type de a_i de t
  - $val(t) = \{(a_1,v_1), (a_2,v_2), \ldots (a_n,v_n)\}$ valeur de t
  - $val(t, a_i) = v_i$ valeur de a_i de t
  - $deg(t) = n$ degré de t
  - $id(t) = \{a_1, a_2, \ldots, a_n\}$ les identifiants d'attributs de t

Observations

- *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 – 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 `compétences` et `disponibilités` sont des ensembles.

1FN – Exemple
Normalisation

- Si une relation E1 n'est pas en 1FN, on la normalise en remplaçant chaque attribut non atomique de E1 par une nouvelle relation E2 dont les attributs sont:
  - Une clé candidate de E1.
  - Les attributs des éléments de l'ensemble représentant l'attribut non atomique.

1FN – Exemple
Normalisation de Cours en 1FN

cours

sigle	titre	préalables
IFT287	Lab. de BD	IFT187
IFT487	BD	IFT287 IFT339

cours

sigle	titre
IFT287	Lab. de BD
IFT487	BD

préalablesCours

sigle	préalable
IFT287	IFT1787
IFT487	IFT287
IFT487	IFT339

1FN – Exemple
Normalisation de Professeur

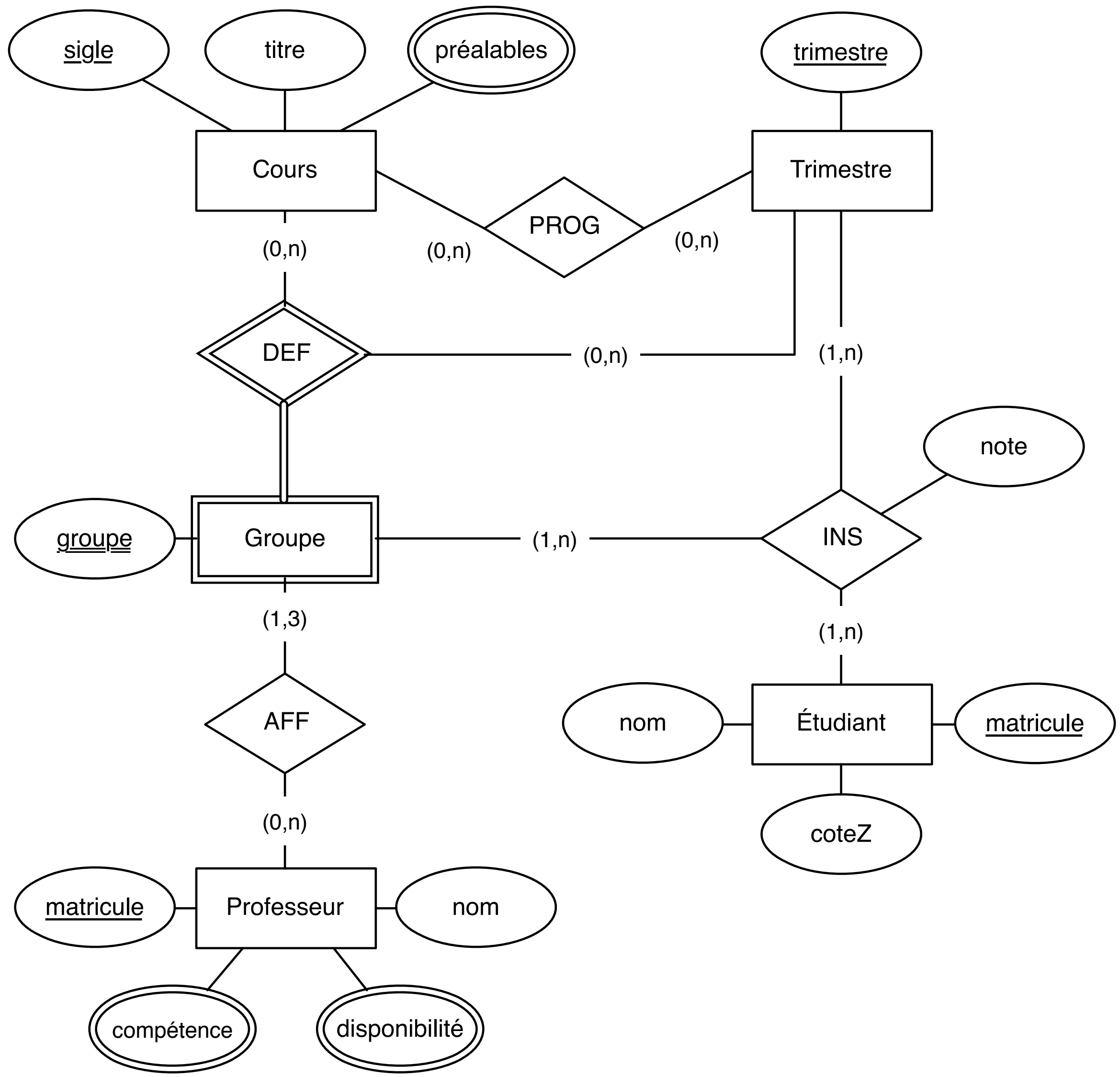
professeur				
matricule	**salaire**	**nom**	**disponibilités**	**compétences**
1	35 000 $	xyz	A01 E02	IFT286 IFT339
2	25 000 $	abc	H01	IFT178

disponibilité	
matricule	**session**
1	A01
1	E02
2	H01

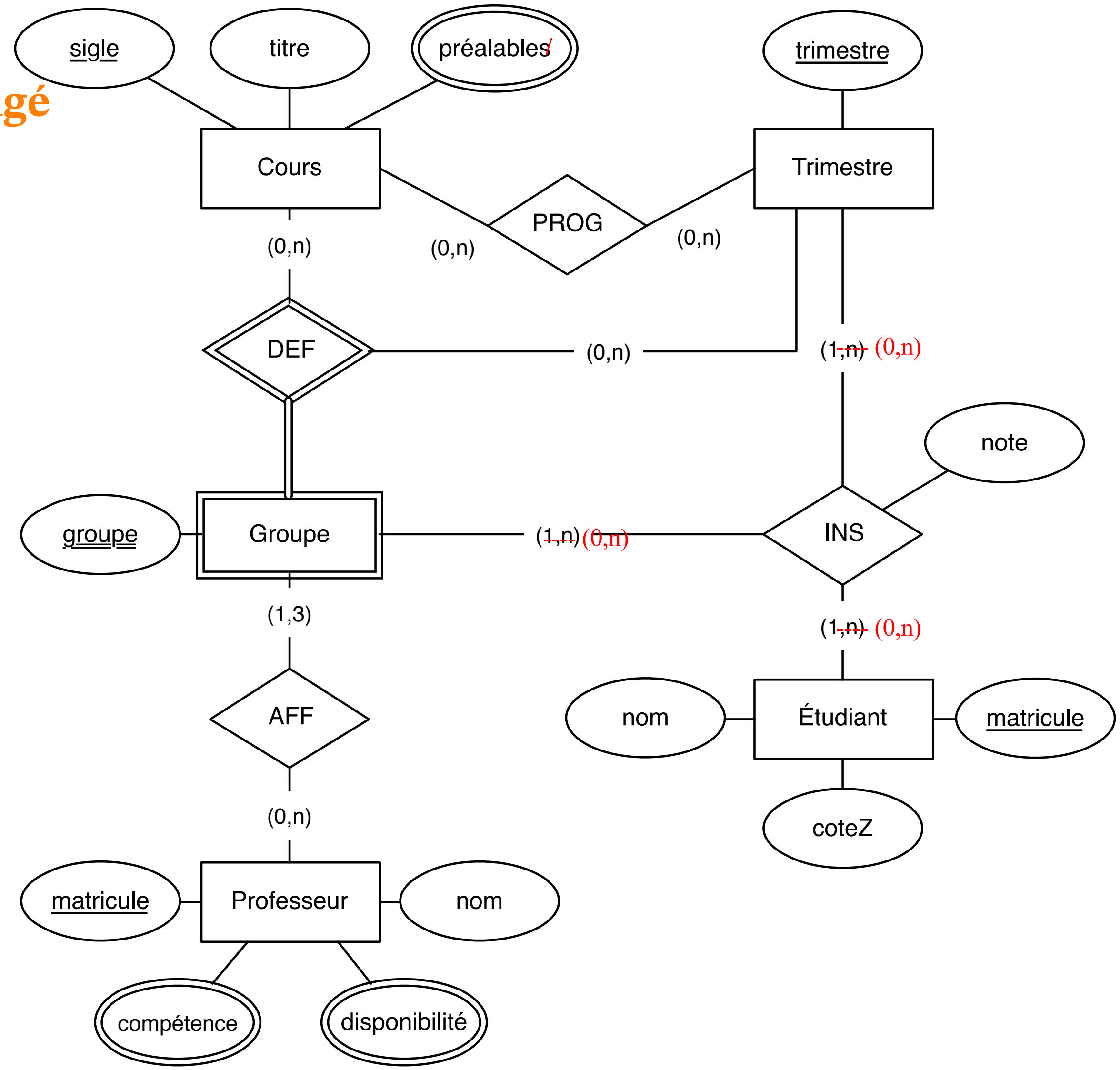
professeur		
matricule	**salaire**	**nom**
1	35 000 $	xyz
2	25 000 $	abc

compétence	
matricule	**sigle**
1	IFT286
1	IFT339
2	IFT178

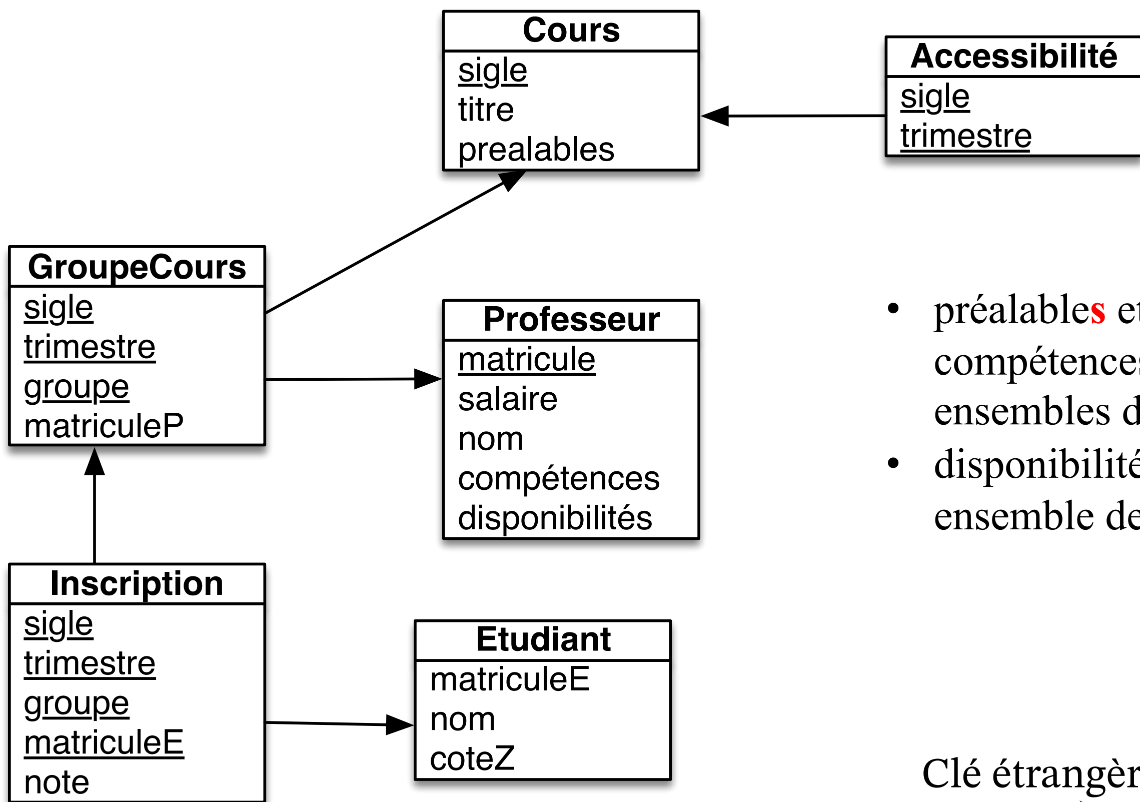
1FN – Exemple
Le MCD des cours



1FN – Exemple
Le MCD des cours — corrigé



1FN – Exemple
Le modèle logique relationnel «naïf» (avant normalisation)

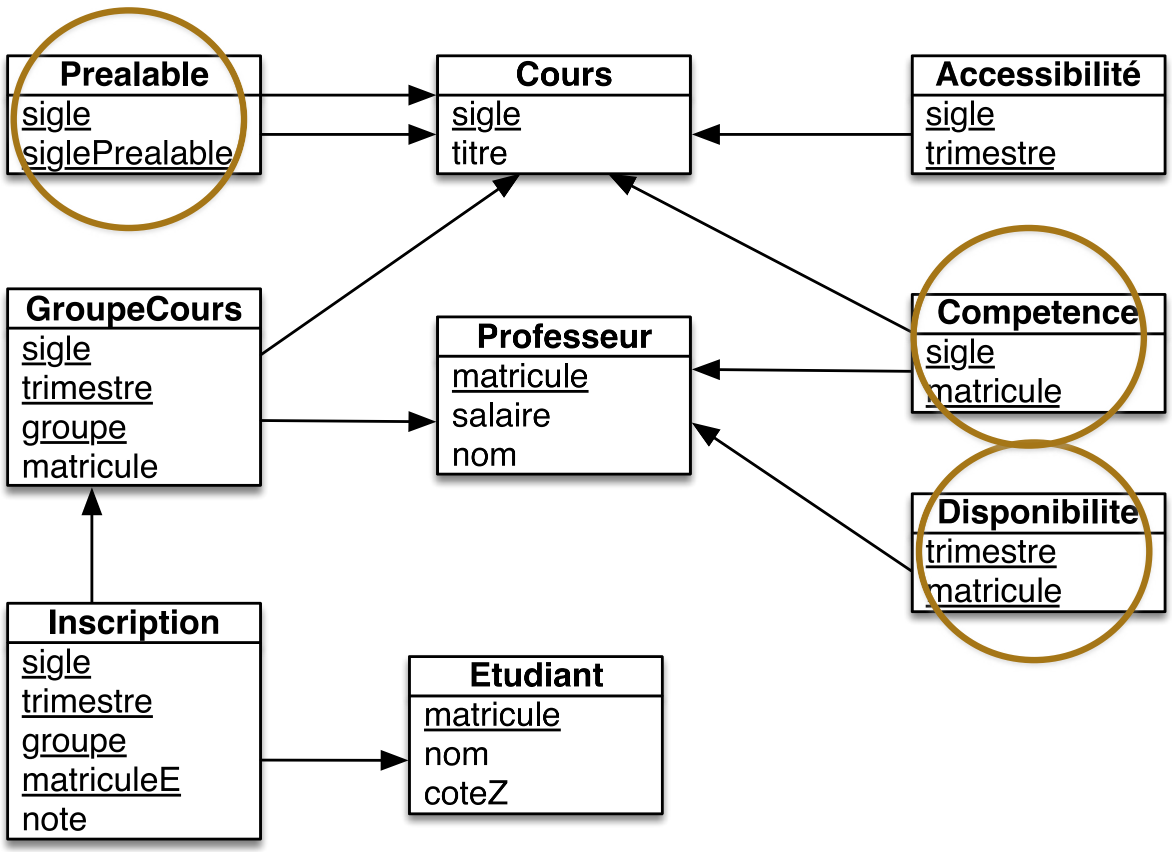


- préalables et compétences sont des ensembles de cours;
- disponibilités est un ensemble de trimestres;

Clé étrangère →

Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

1FN – Exemple
Le modèle logique des cours, après normalisation



1FN
Pourquoi normaliser?

- La théorie relationnelle repose sur l'hypothèse que toute variable est relationnelle et que 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 hypothèse qu'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 atomique dans une BD (et la perte de performance qui en pourrait en découler) est souvent invoquée
- Bien que souvent avérée, elle apparait secondaire en regard de la cohérence, de la validité et de l'efficacité garanties par la 1FN.

1FN
Multiplicité et normalisation

- Certains modèles relationnels permettent d'associer des types non scalaires aux attributs:
  - en maintenant l'hypothèse de base, le modèle relationnel étendu (Tutorial D) autorise les attributs de type relationnel (opérateurs wrap et unwrap);
  - en abandonnant (localement) l'hypothèse de base, le modèle relationnel-objet (les multiset de SQL-1999).

1FN
Multiplicité cachée

- Une erreur fréquente consiste à définir le type d'un attribut comme une chaine de caractères et d'y encoder plusieurs valeurs.
- Exemple au tableau!

Dépendances fonctionnelles

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

Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

Définition (1)

- 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 dépend *fonctionnellement* de X) ≣ (X détermine *fonctionnellement* A)
  - X est le déterminant et A le déterminé.

Définition (2)

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

Définition (3)

- 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.

Définition (4)

- 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 apparait lui aussi en ces mêmes tuples «de façon redondante»).

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

$$\text{matriculeE} \rightarrow \text{nom}$$

- 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.

Dépendances fonctionnelles
Précisions

- La DF [matriculeE → 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.

Irréductibilité et trivialité (1)

- Définitions
  - Une DF [A→B] est **triviale**
    si et seulement si B est un sous-ensemble de A.
  - Une DF [A→B] est **irréductible**
    si et seulement si aucun sous-ensemble propre de A ne détermine B.
  - Une DF [A→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.

Dépendances fonctionnelles
Irréductibilité et trivialité (2)

- Synonymes
  - Une dépendance irréductible est aussi dite complète.

Clé candidate

- S'il existe une DF irréductible entre $\{A_1, ..., A_k\}$ et tous les autres attributs $A_{k+1},..., A_n$ d'une relation R, alors $\{A_1, ..., A_k\}$ est une clé candidate de R.

- Réciproquement, toute clé candidate $\{A_1, ..., A_k\}$ de R définit une DF irréductible vers chacun des autres attributs $A_{k+1},...,$ A_n de R.

- *Notation*
  - *Ces DF sont désignées les DF «directement induites» par (les clés de) R.*

Dépendances fonctionnelles
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 $WX \rightarrow Z$

Comment les découvrir?

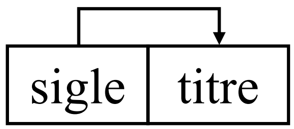
- Les DF sont des propriétés issues du domaine d'application.

- On les détermine à partir de notre connaissance des faits (règles, conditions, etc.) du domaine d'application.

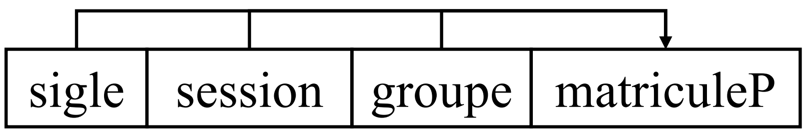
- On peut déterminer s'il y a une dépendance fonctionnelle $\{A_1, \ldots, A_n\} \rightarrow A_{n+1}$ en répondant à la question suivante:
  - Lorsque toutes les valeurs des attributs $A_1, \ldots, A_n$ sont connues, peut-on **toujours** associer une et une seule valeur à l'attribut A_{n+1}?

Dépendances fonctionnelles
Représentation graphique

- sigle → titre



- (sigle, session, groupe) → matriculeP



Définition de la FNBC

- Une relation R est en FNBC relativement à une DF applicable non triviale [X→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→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 clé (de R).*

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

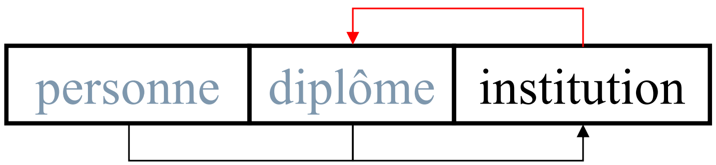
Supposons

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. [institution → diplôme]
2. [personne, diplôme → institution]

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



Cette relation n'est pas en FNBC relativement à la DF [institution → diplôme], car l'institution n'est pas une sur-clé.

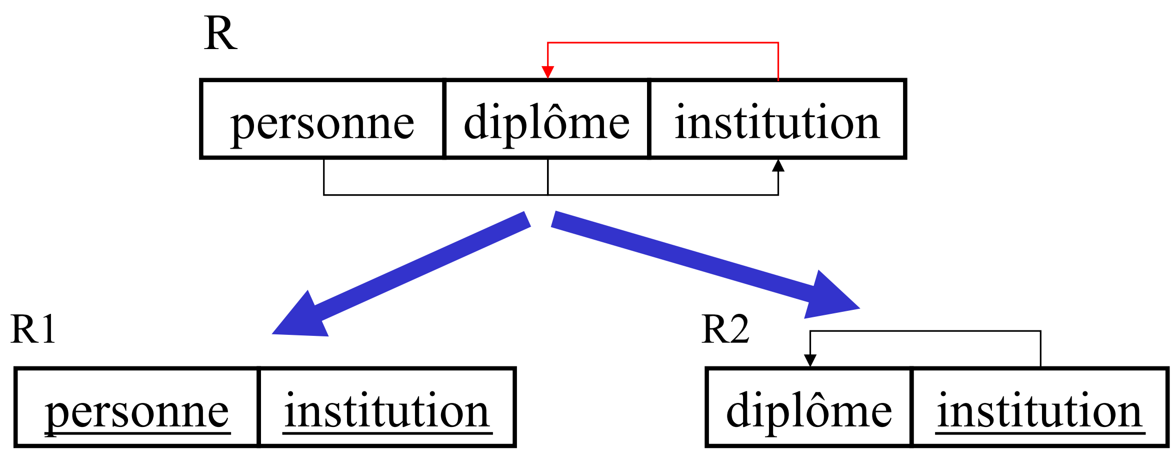
Normalisation en FNBC

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 ⋈ R2) :

#R = #(R π {personne, diplôme})

- Traduire cette contrainte en SQL
  - avec un invariant (ASSERTION):

```
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

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

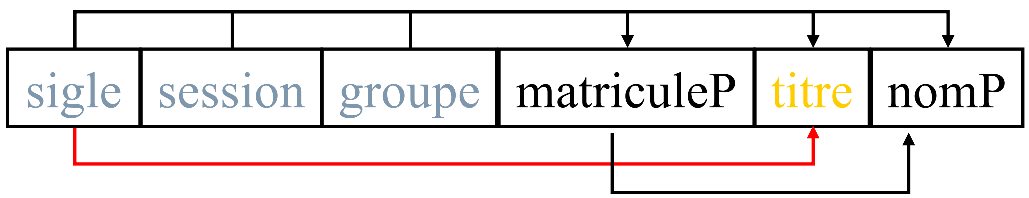
Définition de la 2FN

- 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.

Définition de la 2FN (bis)

- Une relation R est en 2FN si et seulement si, pour chaque DF applicable non triviale [X→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.

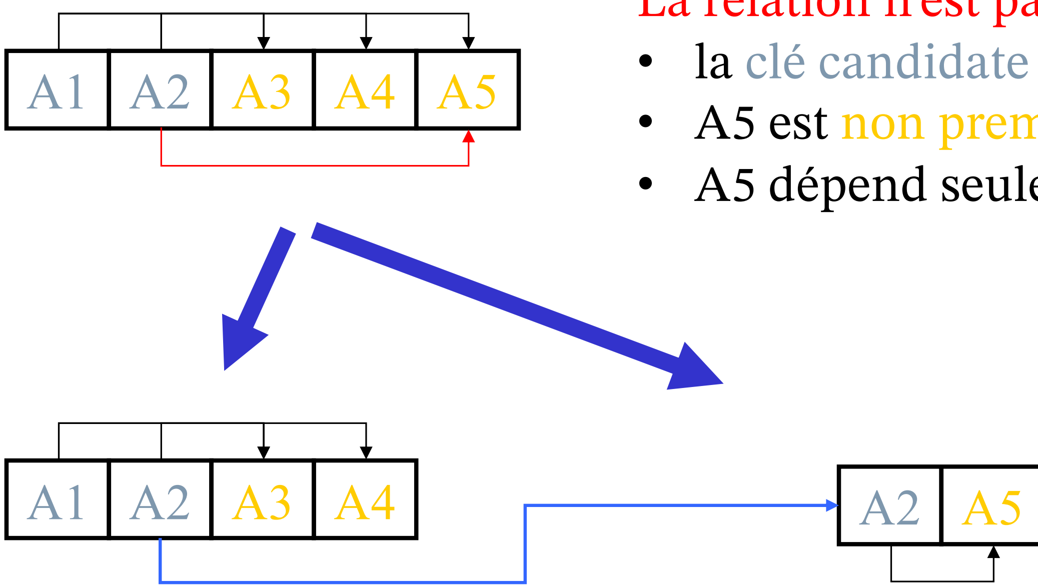
Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

- 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.

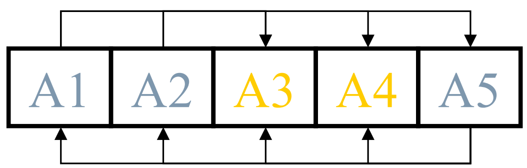
Dépendances fonctionnelles
Exemples de normalisation en 2FN

La relation n'est pas en 2FN, car
- la clé candidate est {A1,A2}
- A5 est non premier
- A5 dépend seulement de {A2}

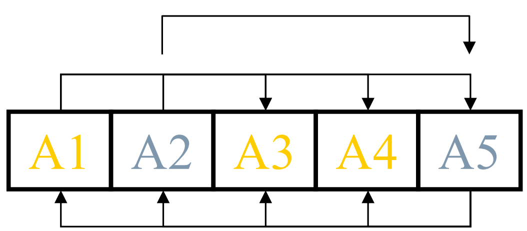


L'ajout d'une clé référentielle complète la garantie d'intégrité.
Est-ce obligatoire? Pourquoi?

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

Définition de 3FN

- Une relation R est en 3FN si et seulement si, pour chaque DF applicable non triviale [X→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.~~

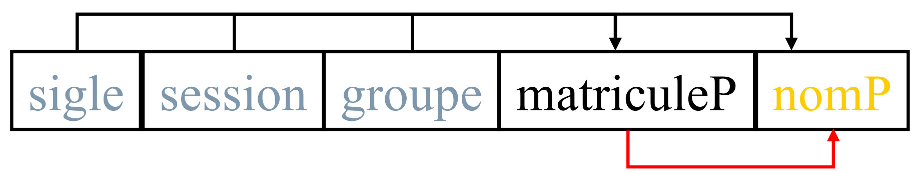
Dépendances fonctionnelles
Contre-exemple de 3FN

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

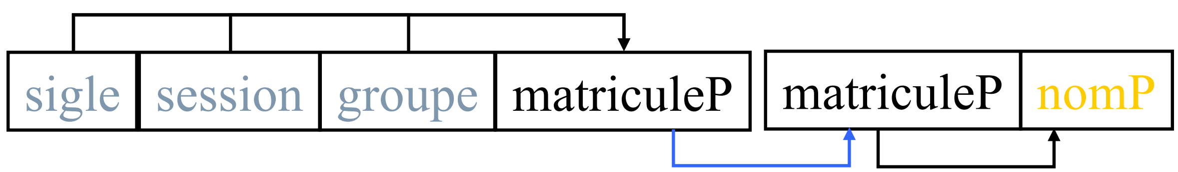
- X est une sur-clé,
- A est un attribut premier.

La relation suivante n'est pas 3FN, car:

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

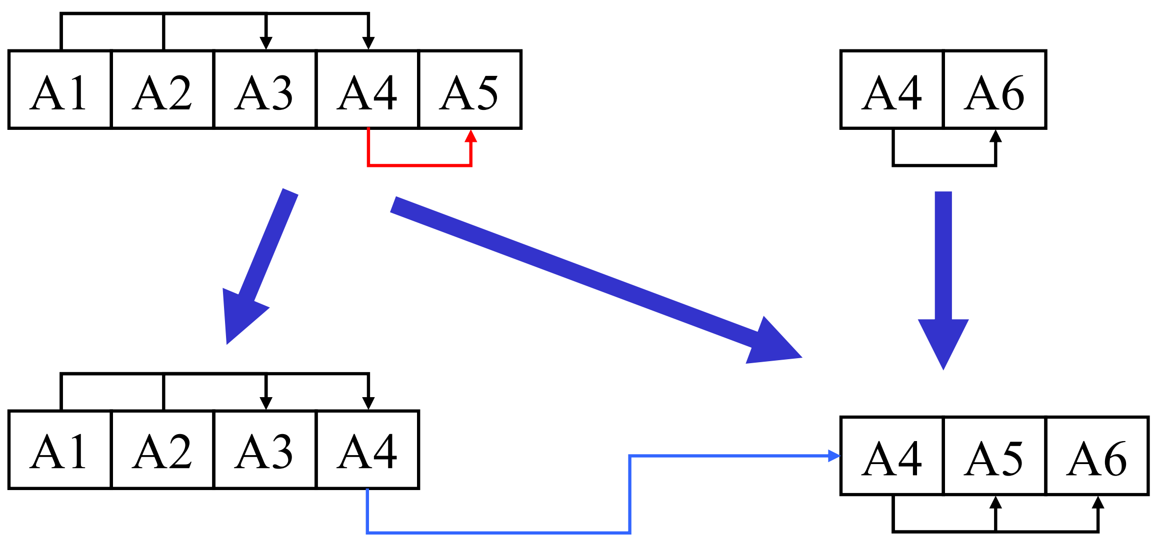


Partition et normalisation 3FN



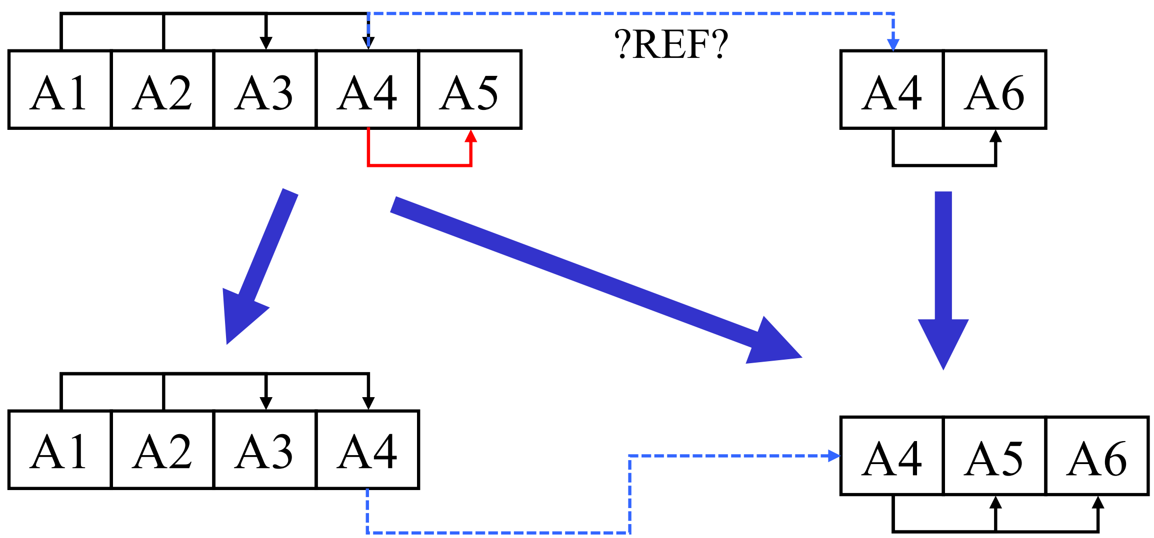
L'ajout d'une clé référentielle complète la garantie d'intégrité.
Est-ce obligatoire? Pourquoi?

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?

Définition de la FNBC (bis)

- Une relation R est en FNBC relativement une DF non triviale [X→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
Discussion (1)

- La FNBC ne peut pas toujours être atteinte sans perte (de DF), il faut donc se rabattre parfois sur la 3FN (ou se résoudre à ajouter des contraintes autres des contraintes de clés internes).

Discussion (2)

- En fait, il vaut parfois la peine de faire l'exercice de comparer les deux solutions, car il peut s'avérer plus facile et plus sûr (voire performant) d'ajouter au modèle logique FNBC les contraintes appropriées que de prendre en compte et contrôler les incohérences dans le modèle logique 3FN (dans toutes les requêtes applicables).
  - Si on conserve la FNBC, la contrainte sera une contrainte de BD faisant intervenir (au moins) deux relvar.
  - Si on se rabat sur la 3FN, la contrainte sera une contrainte (interne) à la relvar, mais insuffisante.
  - En SQL, dans les deux cas, il sera la plupart du temps nécessaire de définir des TRIGGER, faute d'ASSERTION.

2FN, 3FN, FNBC: un résumé

- Soit les conditions suivantes relativement à une DF non triviale [X→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→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 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

Exemple A — une clé (1)

- Soit la relvar
  OffreDeCours {sigle, session, matriculeP}
  représentant le fait qu'un professeur (matriculeP) donne le cours (sigle) à une session (session) donnée.

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

Exemple A — une clé (2)

OffreDeCours peut-elle représenter également les trois relations suivantes?

- disponibilité {matriculeP, session}
  - le professeur peut enseigner durant cette session;
- compétence {matriculeP, sigle}
  - le professeur est apte à enseigner ce cours;
- offre {sigle, session}
  - le cours est offert à cette session.

Ce qui se traduit par OffreDeCours représente-t-elle (toujours et complètement) les trois relations disponibilité, co0mpétence et offre?

Dépendances de jointure
Exemple A — une clé (3)

OffreDeCours n'est pas satisfaisant, car elle n'est pas la jointure des trois autres relations.
Par exemple:

OffreDeCours		
sigle	session	matriculeP
IFT 333	2014-1	1
IFT 333	2014-3	2

$\neq$

Offre	
sigle	session
IFT 333	2014-1
IFT 333	2014-3

$\bowtie$

Disponibilité	
matriculeP	session
1	2014-1
2	2014-1
2	2014-3

$\bowtie$

Compétence	
matriculeP	sigle
1	IFT 333
2	IFT 333

Définition

- $\bowtie\{d_1, \ldots, d_k\}$ est une dépendance de jointure (DJ) sur E, si et seulement si
  - $d_1 \subseteq E, \ldots, d_k \subseteq E$
  - $E = d_1 \cup \ldots \cup d_k$

Notation

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

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
  - $R = (R \pi d_1) \bowtie \ldots \bowtie (R \pi d_k)$

- Une relation est en 5FN si elle l'est relativement à chacune des DJ qui lui sont applicables.

Dépendances de jointure
Exemple A — une clé (4)

- 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.
  - Exercice: identifier chacun de ces prédicats.
  - Question: pourquoi est-ce important?
  - Question: comment changer la modélisation?

Dépendances de jointure
Exemple A — une clé (3bis)

OffreDeCours n'est pas 5FN: les clés de Offre, Disponibilité et Compétence ne sont pas des surclés de OffreDeCours.
Par exemple:

OffreDeCours		
sigle	session	matriculeP
IFT 333	2014-1	1
IFT 333	2014-3	2

$\neq$

Offre	
sigle	session
IFT 333	2014-1
IFT 333	2014-3

$\bowtie$

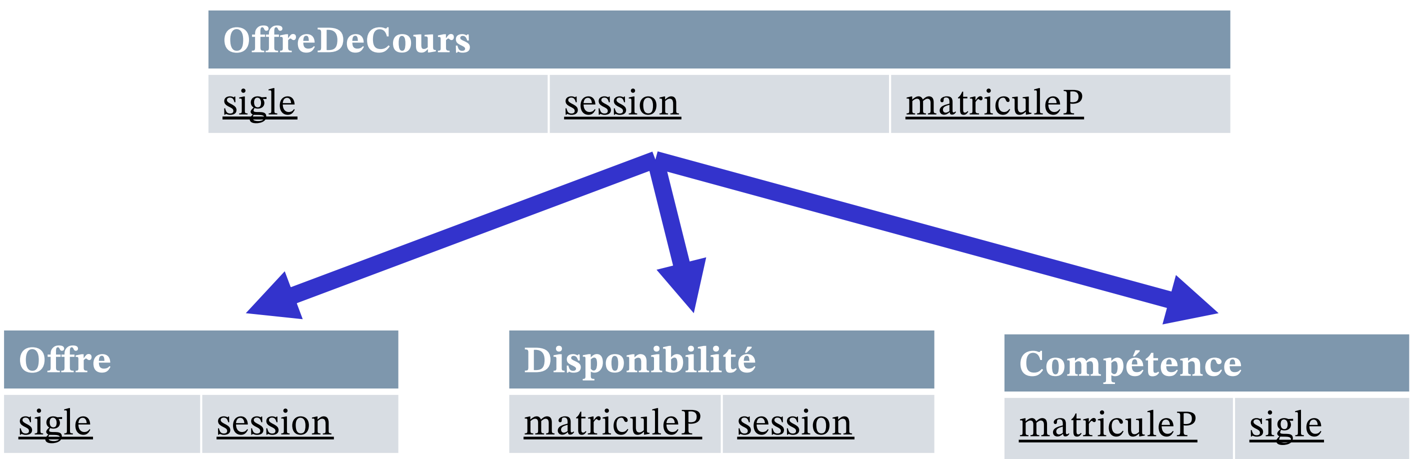
Disponibilité	
matriculeP	session
1	2014-1
2	2014-1
2	2014-3

$\bowtie$

Compétence	
matriculeP	sigle
1	IFT 333
2	IFT 333

Dépendances de jointure
Exemple A — une clé (5)

- C’est-à-dire qu’OffreDeCours ne peut pas représenter les 4 relvars en une!
- Il faudra donc les maintenir et ajouter 3 contraintes référentielles.



Dépendances de jointure
Exemple A — une clé (6)

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 offert* à cette session.
- OffreDeCours {matriculeP, session, sigle}
  - le professeur *enseigne* le cours identifié par sigle durant cette session.

Exemple A — une clé (7)

- 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 cherche la bonne information au bon endroit (donc, connaitre les prédicats).
- Que fera-t-il s'il ne connait que l'entête des relations, mais pas les prédicats?
- Comment qualifieriez-vous un concepteur de BD qui ne documenterait pas toutes ses relvars?

Induction

- Une DJ est induite par les clés de R si et seulement si toute valeur de R conformes aux clés de R vérifie que $R = (R\ \pi\ d_1) \bowtie \ldots \bowtie (R\ \pi\ d_k)$.

- Une DJ est induite par les clés de R si et seulement si la fermeture des composantes de DJ par les clés de R contient l’entête de R.

Dépendances de jointure
5FN – cas général

Une relation R est en cinquième forme normale (5FN) si et seulement si toutes les DJ qui lui sont applicables sont induites par les clés de R.

Dans ce cas, on démontre que

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

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 sont une surclé de R.

- Il s'ensuit la définition de la 5FN dans le cas simple (une seule clé candidate).

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} };

Théorème général de projection-jointure

- Soit R une relvar dont l'entête est E.
- Soit les composants $d_1, \ldots, d_k$ 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 $\bowtie\{d_1, \ldots, d_k\}$.

Dépendances multivaluées

Un cas intéressant des dépendances de jointure

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

Dépendances multivaluées
Exemple 1

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

cours	enseignant	livre
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

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

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

Définition

- Soit X et Y deux sous-ensembles d'attributs d'une relation R, une DM X↠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→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
- $Z = E — (X \cup Y)$

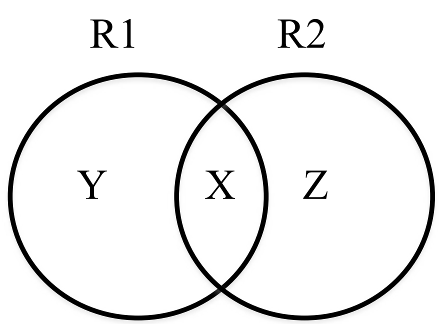
R est en 4FN relativement à la DM $X \twoheadrightarrow Y$ si et seulement si

- $(X \cup Y)$ 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 ∪ R2



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

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

nom	adresse	ville	emploi
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}↠{a,v} (ou de façon équivalente {n}↠{e}).
- On remarque que la DF {n}→{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}→{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

- FNBC + ANC ⇒ 5FN
- 3FN + 0CCC ⇒ 5FN
- RT + 0DFNT ⇔ RT + FNBC
- Rappel : relation totale (RT)
- Synthèse
- Pour aller un peu plus loin… la 6FN

Quelques théorèmes utiles
FNBC + ANC ⇒ 5FN

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

Quelques théorèmes utiles
3FN + 0CCC ⇒ 5FN

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

Quelques théorèmes utiles
RT + 0DFNT ⇔ 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 dépendance non triviale!

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).

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, elle est de degré n, et n'a aucune clé de degré inférieur à n — 1.
- 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 $\{D_1, \ldots, D_n\}$
  - 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 $X \to \{A_1, A_2, \ldots, A_n\}$ in F by the n functional dependencies $X \to A_1, X \to A_2, \ldots, X \to A_n$.
3. For each functional dependency $X \to A$ in F
   for each attribute B that is an element of X
      if $\{ \{F - \{X \to A\} \} \cup \{ (X - \{B\}) \to A\} \}$ is equivalent to F
         then replace $X \to A$ with $(X - \{B\}) \to A$ in F.
4. For each remaining functional dependency $X \to A$ in F
   if $\{F - \{X \to A\} \}$ is equivalent to F,
      then remove $X \to A$ from F.

Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec

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 -> Y_1,…, X -> 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 -> Y une dépendance non satisfaite
      - Remplacer Z par les composantes (X U Y) et (Z — Y)

Normalisation FNBC — Elmasri, p. 559

16.3.2 Nonadditive Join Decomposition into BCNF Schemas

The next algorithm decomposes a universal relation schema $R = \{A_1, A_2, ..., A_n\}$ into a decomposition $D = \{R_1, R_2, ..., R_m\}$ 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 $D = \{R_1, R_2, ..., R_m\}$ 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 -> Y_1,..., X -> 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

Normalisation 3FN – Elmasri, p. 558

16.3.1 Dependency-Preserving Decomposition into 3NF Schemas

Algorithm 16.4 creates a dependency-preserving decomposition $D = \{R_1, R_2, \ldots, R_m\}$ 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 $\{X \cup \{A_1\} \cup \{A_2\} \ldots \cup \{A_k\}\}$, where $X \rightarrow A_1, X \rightarrow A_2, \ldots, X \rightarrow A_k$ 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/

Département d'informatique, Faculté des sciences, Université de Sherbrooke, Québec