Introduction à la normalisation relationnelle

Abdelkrim LAHLOU

Contact.Lahloukarim@gmail.fr

Plan

I. Introduction
II. Dépendance fonctionnelle
III. Formes normales (1FN – 2FN – 3FN)
IV. Algorithmes de normalisation
V. Conclusion

I – Introduction

- La théorie de la normalisation permet de définir une méthode de conception de « bonnes » tables, c'est-à-dire **sans redondance** et **sans perte d'information**
- Exemple :

NumPropriétaire	Nom	Ville	NumVéhicule	Marque	Date
1000	AAAA	PARIS	90FE75	PEUGEOT	10-sep-89
1500	BBBBB	NANTES	43XY97	RENAULT	02-fev-96
1000	AAAA	PARIS	56GT98	FIAT	06-mar-91
1350	CCCC	NICE	43ZT88	RENAULT	28-dec-87
1500	BBBBB	NANTES	57TG92	PEUGEOT	26-jui-91

- *Redondance :* on dit 2 fois que le propriétaire N°1500 a pour nom BBBBB et habite à Nantes

Pourquoi la normalisation ?

- pour éliminer les redondances
- pour mieux comprendre les relations sémantiques entre les données
- pour éviter les incohérences de mise à jour
- pour éviter, autant que possible, les valeurs nulles

  Insertion d'une personne sans voiture $\Rightarrow$ introduction de valeurs nulles
- Pour éviter la perte d'information

  Suppression de la dernière voiture possédée par une personne $\Rightarrow$ perte d'information

Exemple

- ***Relation COURS***

Nomprof	Ville	Département	Nometud	Age	Nomcours	Note
Dupont	Lille	59	Alfred	22	Math	12
Dupont	Lille	59	Arthur	25	Math	05
Martin	Arras	62	Alfred	22	Anglais	18
Martin	Arras	62	Pierre	23	Anglais	11
Dupont	Lille	59	Pierre	23	Anglais	13
Charles	Lille	59	Pierre	23	Anglais	12

- *des données redondantes :* `Dupont à Lille (59)`
- *des risques d'incohérence :* `déménagement de Dupont à Marseille`
- *des valeurs nulles :* représenter un prof qui n'a pas d'étudiant

entraînent des anomalies à l'interrogation

  – ***Problème du choix des relations***

Comment normaliser un schéma relationnel ?

- **Approche par décomposition :**
  - on part d'une table contenant tous les attributs
  - et on décompose jusqu'à ce qu'il n'y ait plus de redondances

- **Approche par synthèse :**
  - à partir de l'ensemble des attributs
  - et des dépendances fonctionnelles
  - on constitue les tables

Exemple de table non normalisée

NumPropriétaire	Nom	Ville	NumVéhicule	Marque	Date
1000	AAAA	PARIS	90FE75	PEUGEOT	10-sep-89
1500	BBBBB	NANTES	43XY97	RENAULT	02-fev-96
1000	AAAA	PARIS	56GT98	FIAT	06-mar-91
1350	CCCC	NICE	43ZT88	RENAULT	28-dec-87
1500	BBBBB	NANTES	57TG92	PEUGEOT	26-jui-91

Exemple de normalisation par décomposition en utilisant les Dépendances Fonctionnelles

NumPropriétaire	Nom	Ville
1000	AAAA	PARIS
1500	BBBBB	NANTES
1350	CCCC	NICE

NumVéhicule	Marque
90FE75	PEUGEOT
43XY97	RENAULT
56GT98	FIAT
43ZT88	RENAULT
57TG92	PEUGEOT

NumPropriétaire	NumVéhicule	Date
1000	90FE75	10-sep-89
1500	43XY97	02-fev-96
1000	56GT98	06-mar-91
1350	43ZT88	28-dec-87
1500	57TG92	26-jui-91

Décomposition sans perte

Jointure

Soient R (A1, A2, …, An) et S (B1, B2, …, Bp) deux relations

La jointure de R et S est la relation T qui a pour attributs l'union des attributs de R et S et pour tuples l'ensemble des tuples construits à partir de R et S sur les valeurs identiques des attributs communs

0n note $T = R \bowtie S$

Exemple

R	Nom	Salaire
	Dupond	10000
	Durand	5400
	Martin	12000

S	Nom	Adresse
	Dupond	Issy
	Durand	Sète
	Martin	Sète

T	Nom	Salaire	Adresse
	Dupond	10000	Issy
	Durand	5400	Sète
	Martin	12000	Sète

Définition

La décomposition de R en R1, R2, …, Rn est sans perte si, pour toute extension de R, on a :

$$R_1 \bowtie R_2 \bowtie .. R_n = R$$

II – Dépendance fonctionnelle

- Soient
  - $R_1(A_1, \ldots, A_n)$ un schéma relationnel
  - X et Y sont deux sous-ensembles de $\{A_1, \ldots, A_n\}$

- On dit que :
  - Y **dépend fonctionnellement de** X ou bien X **détermine** Y, on note ($\mathbf{X} \rightarrow \mathbf{Y}$), si quelle que soit l'instance de R, pour tout tuples T_1, T_2 de R, on a :

    $$T_1[X]=T_2[X] \Rightarrow T_1[Y]=T_2[Y]$$

    avec $T_i[X]$ la valeur de X pour le tuple T_i.

Exemple

- NumPropriétaire → Nom
- NumPropriétaire → Ville
- NumVéhicule → Marque
- NumPropriétaire, NumVéhicule → Date

Remarques :

- Une DF s'applique sur toutes les instances possibles
- Une DF doit être déclarée

Axiomes d'Armstrong

- **Propriétés des dépendances fonctionnelles**
  1. Réflexivité $Y \subseteq X \Rightarrow X \rightarrow Y$
  2. Augmentation $X \rightarrow Y \Rightarrow X,Z \rightarrow Y,Z$
  3. Transitivité $X \rightarrow Y$ et $Y \rightarrow Z \Rightarrow X \rightarrow Z$

  …
- **Conséquences**
  4. Union $X \rightarrow Y$ et $X \rightarrow Z \Rightarrow X \rightarrow Y,Z$
  5. Pseudo-transitivité $X \rightarrow Y$ et $W,Y \rightarrow Z \Rightarrow W,X \rightarrow Z$
  6. Décomposition $X \rightarrow Y$ et $Z \subseteq Y \Rightarrow X \rightarrow Z$

Dépendance fonctionnelle élémentaire

- Dépendance fonctionnelle élémentaire

  $X \rightarrow A$ telle que

  1) A est un attribut unique
  2) $A \notin X$
  3) Il n'existe pas $X' \subseteq X$ tel que $X' \rightarrow A$

- Dans la recherche des DF, on peut se limiter sans restriction aux DFs élémentaires

Exemple

Table COURS (NOMPROF, VILLE, DEPARTEMENT, NOMETUDIANT, AGE, COURS, NOTE)

Dépendances fonctionnelles valides :

NOMPROF ➲ VILLE
NOMPROF NOMETUDIANT ➲ VILLE
NOMETUDIANT ➲ AGE

Dépendances fonctionnelles invalides :

AGE ➲ NOMETUD

Dépendances fonctionnelles élémentaires

NOMPROF ➲ VILLE

VILLE ➲ DEPARTEMENT

NOMPROF ➲ DEPARTEMENT

NOMETUD ➲ AGE

NOMETUD NOMCOURS ➲ NOTE

NOMCOURS ➲ NOMPROF

Autres définitions

$X \rightarrow Y$ est triviale

- Y est un sous-ensemble de X
- A, B $\rightarrow$ B est une DF triviale

$X \rightarrow Y$ est élémentaire et non triviale

- $Y \not\subset X$ et $\forall X' \subset X,\ X' \not\rightarrow Y$
- Ex : NumPropriétaire, NumVéhicule $\rightarrow$ Date

Autres définitions

- **Fermeture d'un ensemble F de DFs :**
  – Ensemble F' de DFs obtenu par applications successives des axiomes d'inférence
- **Fermeture transitive d'un ensemble F de DFs :**
  – Ensemble F^+ de DFs élémentaires obtenues par application des axiomes de transitivité et de pseudo-transitivite
- **Couverture minimale d'un ensemble F de DFs :**
  – Plus petit ensemble de DFs permettant d'obtenir, par applications successives des axiomes d'inférence, la fermeture transitive de F

Exemple

Exemple : *Relation* COURS

(NOMPROF, VILLE, DEPARTEMENT, NOMETUDIANT, AGE, COURS, NOTE)

1. **NOMPROF → VILLE**
2. **VILLE → DEPARTEMENT**
3. **NOMPROF → DEPARTEMENT**
4. **NOMETUDIANT → AGE**
5. **NOMETUDIANT, COURS → NOTE**
6. **COURS → NOMPROF**

n'est pas minimal car 3 est redondante

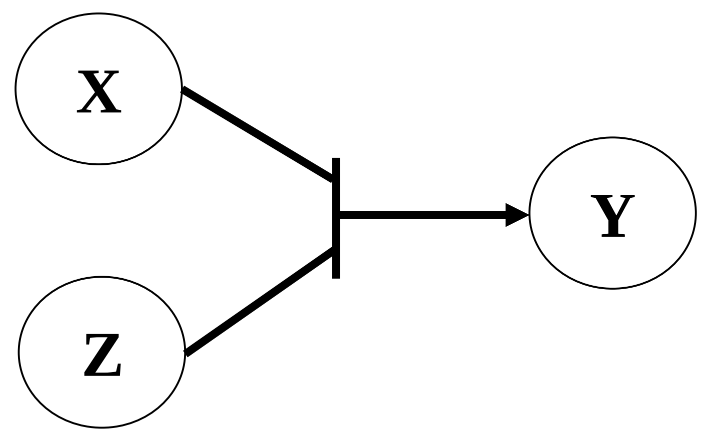
{1, 2, 4, 5, 6} est une couverture minimale

{1, 3, 4, 5} n'est pas une couverture

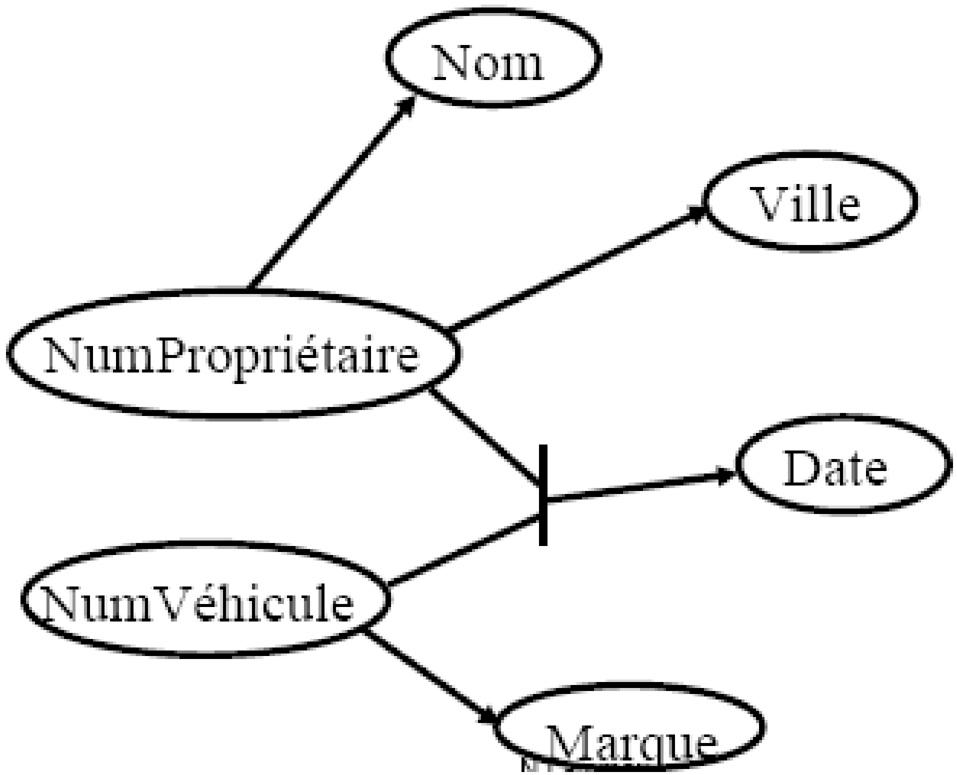
Graphe de dépendance

- **X → Y**

- **X, Z → Y**



Exemple



Dépendance fonctionnelle et Clé

- Soient :

  $R(A1, A2, \ldots, An)$

  $X \subseteq \{A1, A2, \ldots, An\}$

- On dit que X est une clé candidate de R ssi :

  - $X \rightarrow A1, A2, \ldots, An$

  - $\forall\, Y \subset X,\ Y \not\rightarrow A1, A2, \ldots, An$

Exemple

R (NumPropriétaire, NumVéhicule, Nom, Ville, Marque, Date)

- NumPropriétaire → Nom
- NumPropriétaire → Ville
- NumVéhicule → Marque
- NumPropriétaire, NumVéhicule → Date

⬇

{NumPropriétaire, NumVéhicule} est la seule clé pour R

Les Formes Normales

- **1ère Forme Normale (1FN)**
- **2ème Forme Normale (2FN)**
- **3ème Forme Normale (3FN)**
- **Etc, ….**

Première forme normale (1FN)

- *Une relation est en 1ère Forme Normale (1FN) si et seulement si tous ses attributs sont atomiques (non composés et mono-valués)*

- ***Contre-exemples :***

  1. PERSONNE (NOM, PRENOMS)

  Mise en 1FN : PERSONNE1 (NOM, PRENOM1, PRENOM2)

  2. PERSONNE (NOM, PRENOM, ADRESSE)

  Mise en 1FN : PERSONNE2 (NOM, PRENOM, N°RUE, RUE, CODEPOSTAL, VILLE)

Exemple

ETUDIANT (Matricule, Nom , …., DIPLOMES)

01	A	….	{Bac, BTS}
02	B	…..	{Bac, Deug}
03	C	…..	{Bac}

ETUDIANT n'est pas en 1FN

ETUDIANT (Matricule, Nom , …., DIPLOME)

01	A	….	Bac
01	A	….	BTS
02	B	…..	Bac
02	B	…..	Deug
03	C	…..	Bac

ETUDIANT est en 1FN

Deuxième forme normale (2FN)

- *Une relation est en deuxième forme normale (2FN) si :*
  1. elle est en 1 FN
  2. tout attribut n'appartenant pas à la clé dépend uniquement de la totalité de la clé

- *Exemple :*

R ($\underline{A, B}$, C, D) en 1FN et $A \rightarrow C$ $\Rightarrow$ R n 'est pas en 2FN

Exemples de relations non 2FN

R (<u>NumPropriétaire, NumVéhicule</u>, Nom, Ville, Marque, Date

NumPropriétaire ⟶ Nom

NumPropriétaire ⟶ Ville

NumVéhicule ⟶ Marque

NumPropriétaire, NumVéhicule ⟶ Date

R n'est pas en 2FN

Exemple : COURS (NOMPROF, VILLE, DEPARTEMENT, NOMETUD, AGE, NOMCOURS, NOTE)

1 seule clé (NOMETUD, NOMCOURS)

Les DF :

1. NOMPROF ➲ VILLE

2. VILLE ➲ DEPARTEMENT

3. NOMETUD ➲ AGE

4. NOMETUD, NOMCOURS ➲ NOTE

5. NOMCOURS ➲ NOMPROF

Problème pour les attributs NOMPROF, VILLE, DEPARTEMENT et AGE

COURS (<u>NOMETUD NOMCOURS</u> NOTE)
R1 (<u>NOMCOURS</u> NOMPROF VILLE DEPARTEMENT)
R2 (<u>NOMETUD</u> AGE)

sont en 2 FN

Exemple 2FN par décomposition

R (NumPropriétaire, NumVéhicule, Nom, Ville, Marque, Date)

R1 (NumPropriétaire, NumVéhicule, Marque, Date)

R2 (NumPropriétaire, Nom, Ville)

R11 (NumVéhicule, Marque)

R12 (NumPropriétaire, NumVéhicule, Date)

Troisième forme normale (3FN)

- Une relation est en troisième forme normale (3FN) si:
  1. elle est en 2FN
  2. tout attribut n'appartenant pas à une clé ne dépend pas d'un attribut non clé (pas de dépendance fonctionnelle entre attributs non clés)

- Exemple :

R ($\underline{A}$, C, D) en 2FN et C $\rightarrow$ D $\Rightarrow$ R n'est pas en 3FN

Exemple

PRODUIT (NunProduit, Désignation, CodeTVA, TauxTVA)

CodeTVA $\rightarrow$ TauxTVA

PRODUIT (NunProduit, Désignation, CodeTVA)

TVA (CodeTVA, TauxTVA)

Algorithme de mise sous 3 FN

- 0FN $\Rightarrow$ 1FN : mise sous forme atomique des attributs
- 1FN $\Rightarrow$ 2FN : pour chaque partie X de clé déterminant des attributs non clés Y1, …, Yn
  1. on crée une relation supplémentaire avec X pour clé et Y1, …, Yn comme attributs non clés
  2. on retire Y1, …, Yn de la relation initiale
- 2FN $\Rightarrow$ 3FN : pour chaque attribut non clé Y déterminant des attributs non clés Z1, …, Zn
  1. on crée une relation R' supplémentaire avec Y comme clé et Z1, …, Zn comme attributs non clés
  2. on retire Z1, …, Zn de la relation initiale

R' n'est pas nécessairement en 3 FN. Si c'est le cas, réitérer le processus sur R'.

Propriétés

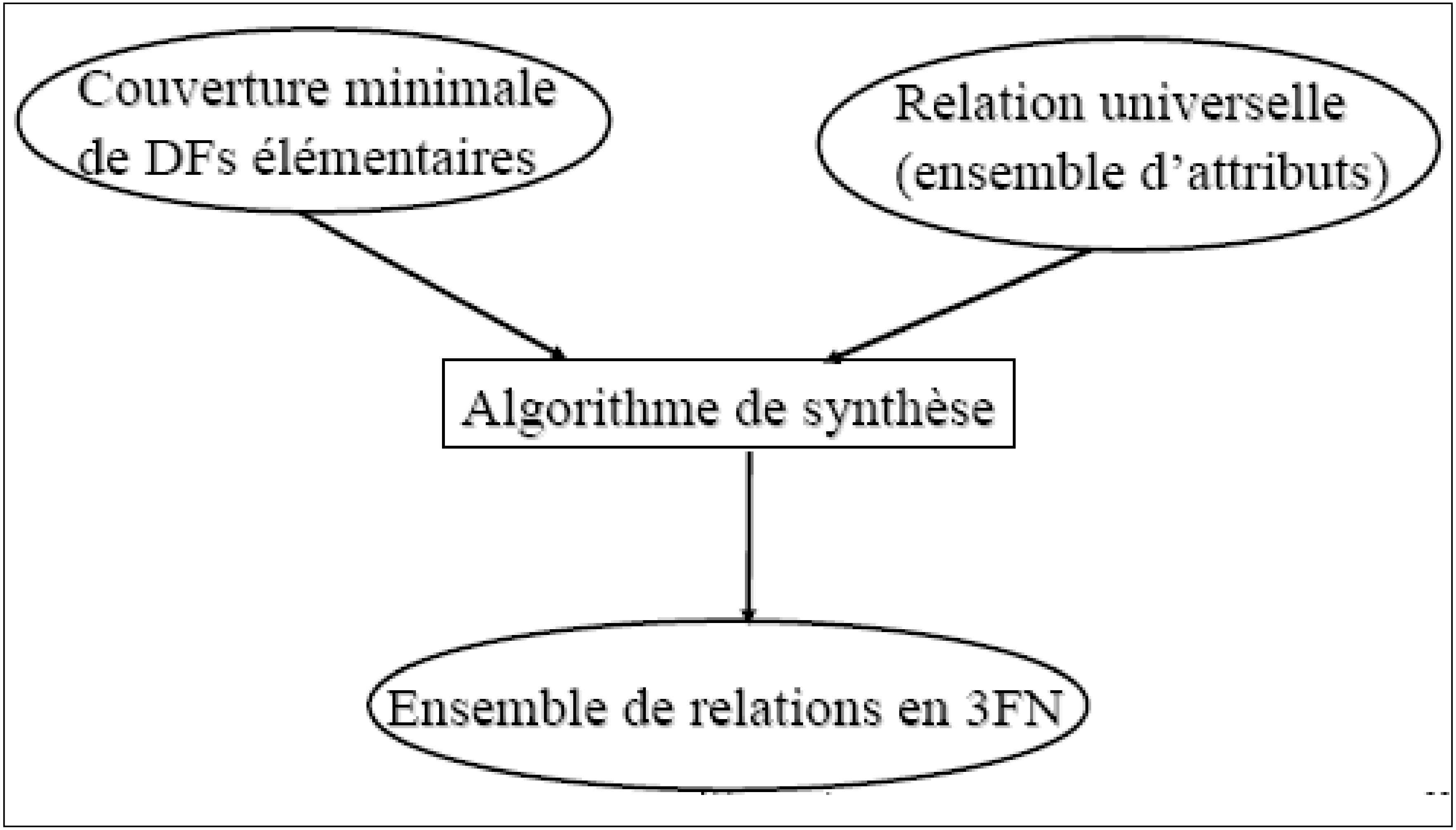
- Dans une décomposition d'une relation en plusieurs autres, on dit que la décomposition préserve une dépendance fonctionnelle s'il reste, après décomposition, une relation contenant tous les attributs de la DF
- **Propriété :**

  Toute relation a au moins une décomposition en 3 FN qui :

  – préserve une couverture minimale de DF

  – est sans perte

Autre algorithme de normalisation

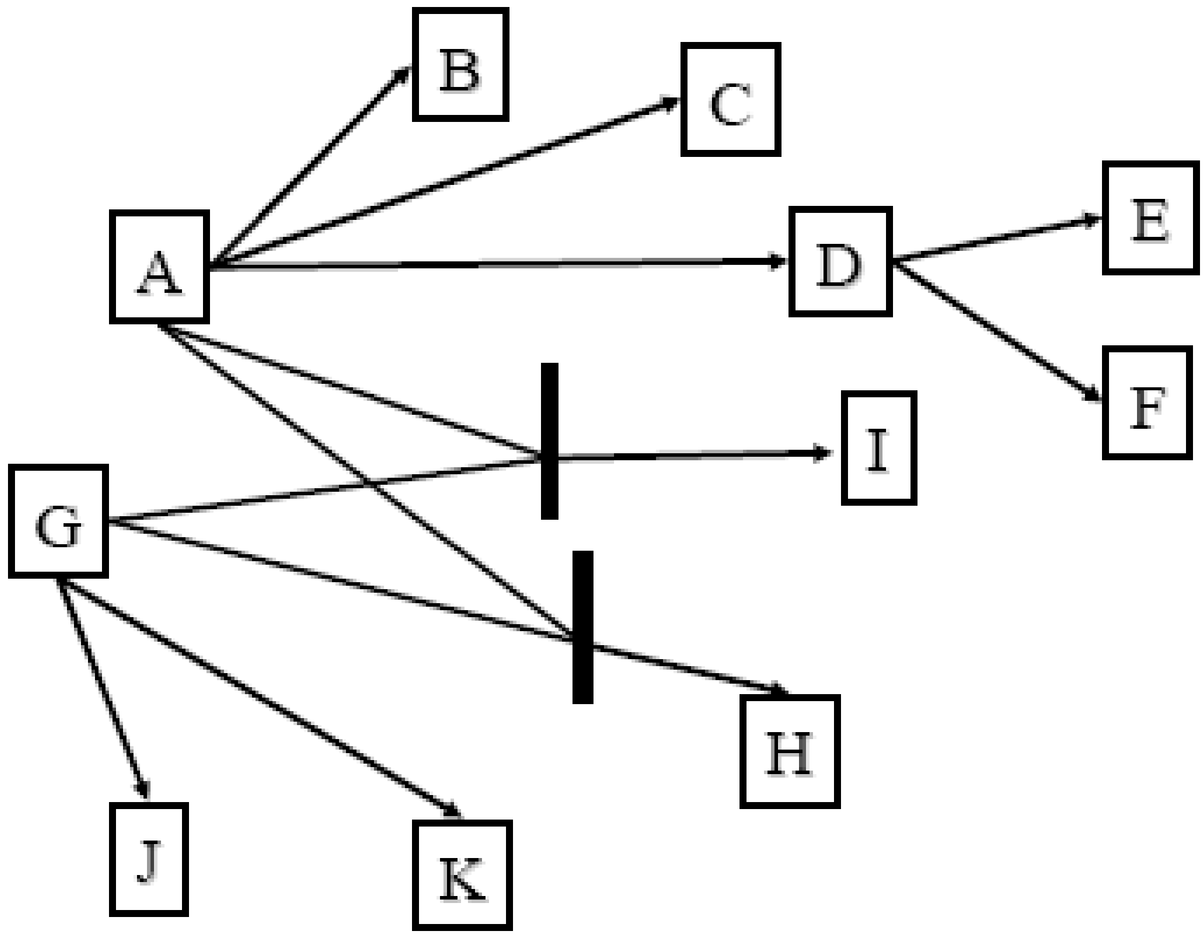


Etapes de l'Algorithme de Synthèse

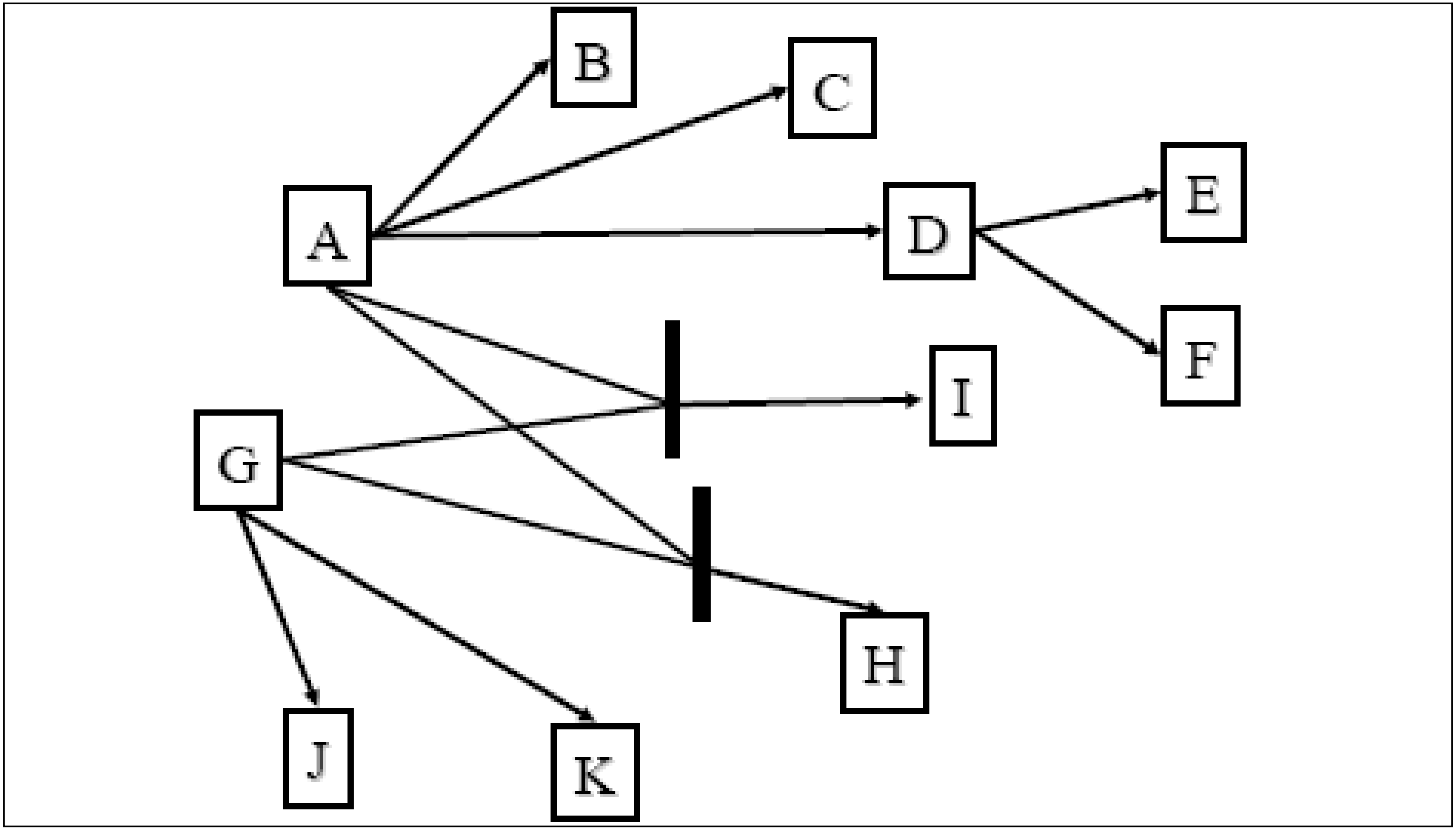
1. **Regroupement des dépendances de même partie gauche**

2. **Construction d'une relation pour chaque ensemble**
   - **Chacune des relation a pour clé le groupe d'attributs en partie gauche**

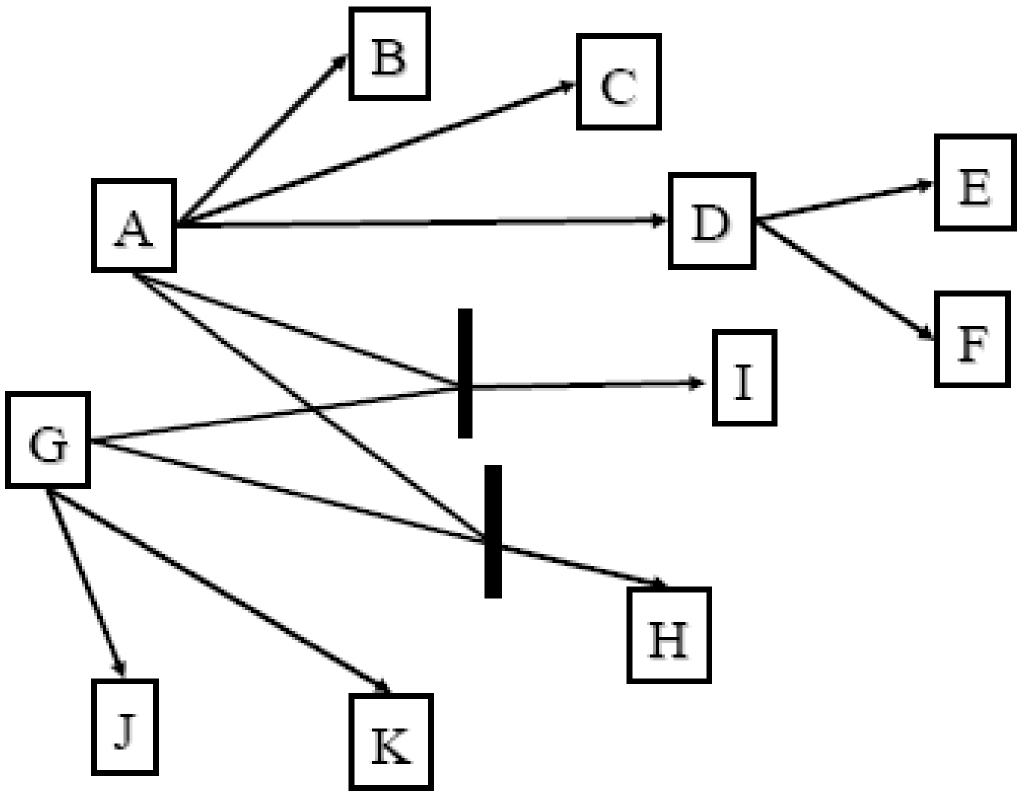
Exemple



Application de l'étape 1 de l'algorithme



Application de l'étape 2 de l'algorithme



R1(**<u>A</u>**, B, C, #D)

R2 (**<u>D</u>**, E, F)

R3(#**<u>A, #G</u>**, H, I)

R4(**<u>G</u>**, J, K)

Clé primaire en gras souligné

Clé étrangère précédée de #

Comparaison des 2 algorithmes

- Algorithme de décomposition :
  - préserve le contenu
  - Conduit à des relations en au moins 3FN

- Algorithme de synthèse
  - préserve les DFs
  - conduit à des relations en 3FN

- **NB :** une décomposition de R en R1, R2, …Rn préserve le contenu ssi la jointure des relations de R1, R2, …Rn est égale à la relation R

CONCLUSION

- La normalisation permet de :
  - Construire des tables sans redondance
  - Vérifier la bonne conception des tables issues de la modélisation conceptuelle
  - Restructurer une base existante