Département Informatique

Module	Bases de Données	
Niveau	Licence 2	2023/2024

Chapitre 3

CONCEPTION DES BASES DE DONNEES.
Modèle Relationnel.

Le modèle relationnel a été défini par E.F Codd dans les années 70 et de nombreux chercheurs ont contribué à son développement. Les premiers SGBD bâtis sur ce modèle ont été SQL/DS et DB2 de IBM, d'où est né le langage de manipulation de bases relationnelles, SQL (Structured Query Language).

Une base relationnelle est composée de tables et est perçue par l'utilisateur comme un ensemble de tables et rien d'autre. Dans une table, une ligne correspond à un enregistrement et une colonne à un champ de cet enregistrement.

Toute opération relationnelle sur une table génère une nouvelle table, c'est-à-dire fonctionne sur un ensemble de données sans que l'on ait à se préoccuper de traiter successivement chacune des données récupérées par l'opération.

I. CONCEPTS DE BASES

I.1. DOMAINE

Un domaine est un ensemble de valeurs atomiques ⟹Un domaine est un ensemble de valeurs que peut prendre un attribut ; c'est le domaine définition d'un ou plusieurs attributs.

Exemples :

- ✓ Dnom : chaînes de caractères de longueur maximale 30
- ✓ Dnum : entiers compris entre 0 et 99999
- ✓ Dcouleur : {Bleu, Blanc, Rouge}
- ✓ Les SGBD usuels offrent des domaines prédéfinis standard, comme INTEGER, CHAR, VARCHAR, DATE…

I.2. PRODUIT CARTESIEN

Le produit cartésien $D_1 * D_2 * \times D_n$ est l'ensemble des *tuples* (*n-uplets*) $<V_1,V_2,....V_n>$ tel que ∀ i V_i appartient à D_i.

Exemple :

- ✓ D_1 = {Bleu, Blanc, Rouge} , D_2 = {Vrai, Faux}.
- ✓ $\mathbf{D_1 \times D_2}$ = {<Bleu, Vrai>; <Bleu, Faux>; <Blanc, Vrai>; <Blanc, Faux>; <Rouge, Vrai>; <Rouge, Faux>}

I.3. RELATION

Une relation est un *sous-ensemble nommé* du produit cartésien d'une liste de domaines. Elle est notée $\boldsymbol{R(A_1:D_1,\ldots,A_n:D_n)}$ où $D_1,\ldots,D_n$ sont des domaines. ⟹ Une relation est définie par :

- ✓ Son nom
- ✓ Liste de couples décrivant ses attributs (nom d'attribut : domaine)
- ✓ Sa (ses) clé(s)
- ✓ Sa définition (phrase en français)

Les trois premières informations : nom de la relation, liste des couples (attribut : domaine) et Identifiant(s) constituent le *schéma de la relation.*

Exemple : schéma de la relation Etudiant :

Etudiant (N° Etud : Dnum, Nom : Dnom, Prénom : Dnom, Age : Dâge)

***Note** : On peut également noter la relation sans mentionner les domaines : $\boldsymbol{R(A_1,A_2,\ldots,A_n)}$*

Exemple :
D_1 = COULEUR et D_2 = BOOLEEN
Couleur_Véhicule (Couleur : D_1, Existe : D_2)

Couleur_Véhicule	Couleur	Existe
	Bleu	FAUX
	Blanc	VRAI
	Rouge	VRAI

Remarques :

- ✓ La **population** d'une relation est constituée de l'ensemble des *tuples* de la relation. C'est un ensemble au sens mathématique du terme : il n'y a donc ni doubles, ni ordre (les nouveaux *tuples* sont rajoutés à la fin de la relation).
- ✓ On appel *schéma relationnel* l'ensemble des schémas de relation qui définit une base de données.
- ✓ On appelle *schéma d'une base de données relationnelle* l'ensemble des *schémas de ses relations*.
- ✓ On appelle *base de données relationnelle*, l'ensemble des populations de toutes ses relations i.e. l'ensemble des *schémas de relation* (schémas relationnel) et dont les occurrences sont les tuples de ces relations.
- ✓ Un schéma de relation définit *l'intension de la relation*, exemple : Véhicule (Matricule : entier, Marque : Chaine, Puissance : entier, Couleur : chaine) alors qu'une *instance* de table représente une *extension de la relation* : (Matricule=452510915, Marque =Peugeot 607, Puissance = 7, Couleur : ‘’noire ‘’).

I.4. ATTRIBUT

Un attribut est un nom donné à une colonne d'une relation, il prend ses valeurs dans un domaine.⇨ un attribut est une colonne d'une relation caractérisé par un nom. **Exemple :** Nom, Age et Marié sont des attributs de la relation Personne.

- ✓ Les attributs d'une relation doivent être différents et plusieurs attributs peuvent être définis sur le même domaine.

I.5. CLE D'UNE RELATION

Soient une relation $\mathcal{R}(A_1, A_2,..., A_n)$ et K un sous-ensemble de $A_1,A_2,... ,A_n$. K est une clé de $\mathcal{R}$ si et seulement si : $K \rightarrow A_1, A_2,...,A_n$ et il n'existe pas de sous-ensemble X *inclus dans* K tel que $X \rightarrow A_1,A_2,...,A_n$.

Une clé est donc un ensemble minimum d'attributs d'une relation qui détermine tous les autres. Si une relation comporte *plusieurs clés*, celles-ci seront dites *clés candidates* ou *clés possibles*. En générale on choisira une en particulier que l'on désignera par *clé primaire*.

- ✓ C'est un groupe d'attributs minimum (composée d'un ou plusieurs attributs) qui détermine un *tuple* unique dans une relation.⇨ Il n'existe pas des *tuples* ayant même valeur pour cette clé.

Note : *Toute relation doit posséder au moins une clé.*

Exemples :

Considérons le schéma de la relation suivante : $\mathcal{R}$(A, B, C, D, E). Cette relation est définie en *extension* par les tuples suivants :

A	B	C	D	E
a_1	b_2	c_2	d_3	e_2
a_1	b_2	c_2	d_1	e_4
a_2	b_3	c_2	d_1	e_5
a_2	b_4	c_5	d_1	e_5

Les dépendances fonctionnelles qu'on peut déduire sont : {B→A, E→D, E→A, B→C }⇨ Ainsi, la clé de cette relation est : {B, E}

Remarques :

- ✓ Toutes les clés *candidates* sont des *clés*, pas seulement la clé primaire.
- ✓ Toute clé candidate détermine les autres clés candidates, puisque qu'une clé détermine tous les attributs de la relation.
- ✓ Étant donné qu'une relation dispose forcément d'une clé, si une relation R n'admet aucune clé K sous ensemble des attributs $A_1...A_n$ de R, alors c'est que $K=A_1...A_n$ (la clé est composée de *tous* les attributs de $\mathcal{R}$). On parle de relation "toute clé".
- ✓ Par convention, l'attribut (ou les attributs) constituant la (ou un des) clé(s) est *souligné*.

I.6. CLE ETRANGERE

Groupe d'attributs devant apparaître comme clé dans une autre relation. Les *clés étrangères* définissent les contraintes d'intégrité référentielles suivantes :

- Lors d'une insertion, la valeur des attributs doit exister dans la relation référencée ;
- Lors d'une suppression dans la relation référencée les *tuples* référençant doivent disparaître ;
- Elles correspondent aux liens entité-association obligatoires.

Exemple :

PERSONNE (**NSS**, NOM, PRENOM)
VOITURE (**MODELE**, ANNEE, COULEUR, TYPE, ***NSS***)

CLES ETRANGERES : VOITURE.NSS **REFERENCE** PERSONNE.NSS

II. LES DEPENDANCES FONCTIONNELLES

La notion de dépendance fonctionnelle permet d'établir *des liens sémantiques* entre attributs ou groupe d'attributs. On dit qu'il existe :

- ✓ Une dépendance fonctionnelle de *X* vers *Y* ou,
- ✓ X détermine Y ou que,
- ✓ Y dépend fonctionnellement de X et

On note X → Y où X est dit *source* de la dépendance fonctionnelle et Y sa *destination*. ⟹ X → Y *signifie que si on connaît la valeur de X, on peut connaître celle de Y (X détermine Y ou Y a une dépendance fonctionnelle de X)*

Exemple :
Véhicule (**Matricule**, Marque, Puissance, Couleur)
Matricule →Marque, Puissance, Couleur

Note : *Une DF est définie sur l'intension du schéma et non son extension. Une DF traduit une certaine perception de la réalité.*

II.1. DEFINITION

Étant donné une relation R, nous disons qu'il y a dépendance fonctionnelle (DF) X de R vers Y de R si à *une valeur de X est associée au plus une valeur de Y.*
Question à se poser : connaissant la valeur de X, puis-je connaître la valeur de Y ?

Exemple :
En examinant l'instanciation de la relation R, on peut y découvrir quelques dépendances fonctionnelles.

A → C est vérifiée
C → A n'est pas vérifiée car $C_2 \rightarrow A_2, A_3$
AB→D est vérifiée

Relation R :

A	B	C	D
a1	b1	c1	d1
a1	b2	c1	d2
a2	b2	c2	d2
a2	b3	c2	d3
a3	b3	c2	d4

Remarque : Pourquoi trouver les DF ?

Les DF font partie du schéma d'une BD, en conséquence, elles doivent être déclarées par les administrateurs de la BD et être contrôlées par le SGBD. De plus l'identification des DF est la base indispensable pour déterminer dans quelle forme normale est une relation et comment en diminuer la redondance.

II.2. LES PROPRIETES DES DEPENDANCES FONCTIONNELLS

II.2.1. DEFINITION DE L'INFERENCE

Si F est une *couverture fonctionnelle* de la relation $\mathcal{R}$, On dit qu'une dépendance fonctionnelle : $X \rightarrow Y$ est inférée (déduite) de F si $X \rightarrow Y$ est valide pour tout *tuple* d'une *extension* de $\mathcal{R}$.

II.2.2. LES AXIOMES D'ARMSTRONG

Les Axiomes d'Armstrong sont un ensemble de règles qui permettent d'effectuer des inférences (inductions) de dépendances fonctionnelles à partir d'autres dépendances fonctionnelles.

- **Réflexivité**
  - ✓ *Si Y est inclus dans X ($Y \subseteq X$), alors $X \rightarrow Y$*
- **Augmentation**
  - ✓ *Si $X \rightarrow Y$ et W est un ensemble d'attributs, alors nous pouvons dire $WX \rightarrow WY$*
- **Transitivité**
  - ✓ *Si $X \rightarrow Y$ et $Y \rightarrow Z$, alors $X \rightarrow Z$*

Note : Il est prouvé que les règles d'Armstrong *sont fondées et complètes*. Ce qui signifie que toute dépendance fonctionnelle déduite en utilisant la réflexivité, l'augmentation ou la transitivité est une *dépendance fonctionnelle inférée* et que toute dépendance fonctionnelle qui peut être inférée de F, peut l'être en utilisant seulement les axiomes mentionnés ci-dessus. Il existe cependant d'autres règles d'inférences qu'on peut utiliser :

- **Pseudo transitivité**
  - ✓ *Si $X \rightarrow Y$ et $YW \rightarrow Z$, alors $XW \rightarrow Z$*

  Démonstration :

  1. $X \rightarrow Y$ par augmentation $XW \rightarrow YW$

  2. $XW \rightarrow YW$, $YW \rightarrow Z$ par transitivité $XW \rightarrow Z$
- **Union**
  - ✓ *Si $X \rightarrow Y$ et $X \rightarrow Z$, alors $X \rightarrow YZ$*

  ***Démonstration** :*

  1. $X \rightarrow Y$ par augmentation $XZ \rightarrow YZ$

  2. $XZ \rightarrow YZ$, $X \rightarrow Z$ par pseudo-transitivité $X \rightarrow YZ$
- **Décomposition**
  - ✓ *Si $X \rightarrow YZ$, alors $X \rightarrow Y$ et $X \rightarrow Z$*

  Démonstration :

  1. D'après la réflexivité $YZ \rightarrow Y$ et $YZ \rightarrow Z$

  2. $X \rightarrow YZ$ et $YZ \rightarrow Y$ par transitivité $X \rightarrow Y$

  3. $X \rightarrow YZ$ et $YZ \rightarrow Z$ par transitivité $X \rightarrow Z$

II.3. DEPENDANCES FONCTIONNELLE PARTICULIERES

II.3.1. DF ELEMENTAIRES (DFE)

C'est une dépendance fonctionnelle de la forme $X \rightarrow Y$, où Y est un attribut unique tel que $Y \notin X$ ou $\nexists X' \subset X$ tel que $X' \rightarrow Y \Longrightarrow$ Une dépendance fonctionnelle $X \rightarrow Y$ est *dite élémentaire*, s'il n'existe pas X', inclus dans X, qui assure lui-même une dépendance fonctionnelle $X' \rightarrow Y$.

Exemple :

```
1. Ref_Article → Nom_Article
2. Num_Facture, Ref_Article → Qte_Article
3. Num_Facture, Ref_Article → Nom_Article
```

Les dépendances fonctionnelles 1 et 2 sont élémentaires alors que la 3 ne l'est pas. Parce qu'il existe une partie de la source de la df qui assure elle-même la dépendance fonctionnelle : c'est Ref_Article au niveau de la df_1.

II.3.2. DF DIRECTES (DFD)

Une dépendance fonctionnelle X →Y est dite *directe,* s'il n'existe pas un attribut Z qui engendrerait une dépendance fonctionnelle transitive X → Z → Y.

Exemple :

```
1. Num_Facture → Num_Représentant     VRAI
2. Num_Représentant → Nom_Représentant    VRAI
3. Num_Facture → Nom_Représentant     FAUX
```

Les dépendances fonctionnelles 1 et 2 sont directes alors que la troisième ne l'est pas parce qu'elle peut être déduite par transitivité à partir des deux premières.

II.3.3. DF TRIVIALES (DFT)

Une dépendance fonctionnelle est dite *triviale,* s'il est impossible qu'elle ne soit pas satisfaite. Une dépendance fonctionnelle est triviale si et seulement si le membre droit (destination) est un sous-ensemble du membre gauche (source).

$A_1, A_2, \ldots, A_n \rightarrow B_1, B_2, \ldots, B_m$ **est :**

- ✓ *Triviale* **:** Si $B_1, B_2, \ldots, B_m$ est un sous-ensemble de $A_1, A_2, .., A_n$
- ✓ *Non triviale* : Si au moins un B_i n'appartient pas à $A_1, A_2, .., A_n$
- ✓ *Complètement non triviale* : Si aucun des B_i n'appartient à $A_1, A_2, \ldots A_n$

II.3.4. DF CANONIQUE (DFC)

Une DF X→ Y est dite *canonique* si ça partie droite est réduite à un seul attribut. Toute DF qui n'est pas canonique peut être transformées par décomposition en DFs canoniques. Par exemple : `NumCli→NomCL, PrénomCL, AgeCL` n'est pas canonique, par décomposition, elle donne trois DFs canoniques :

- ✓ `NumCli →NomCL`
- ✓ `NumCli →PrénomCL`
- ✓ `NumCli →AgeCL`

II.4. COUVERTURE FONCTIONNELLE

On appelle *couverture fonctionnelle* d'une relation *l'ensemble des dépendances fonctionnelles* qui lui sont associées.

Exemple :

- F={ `Ref_Article → Nom_Article ; Num_Facture, Ref_Article →Qte_Article ; Num_Facture, Ref_Article → Nom_Article}`

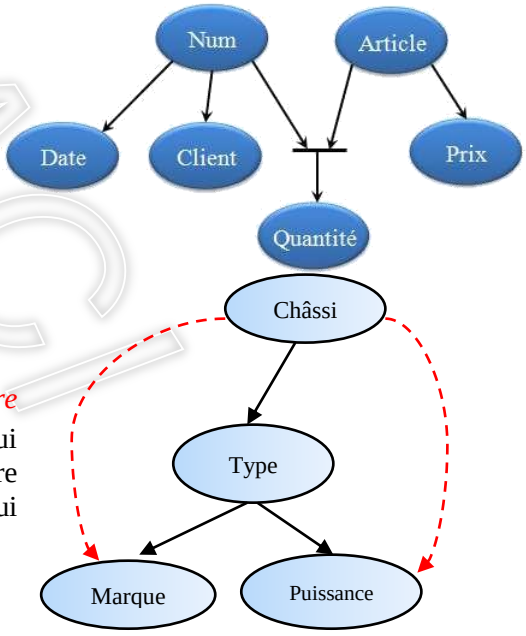
II.5. GRAPHE DE DEPENDANCES FONCTIONNELLES

Un ensemble F de dépendances fonctionnelles (couverture fonctionnelle) peut être représenté avec un graphe orienté où chaque nœud représente un attribut et les arcs les dépendances fonctionnelles entre les attributs.

Exemple :

Commande (Num, Date, Client, Article, Prix, Quantité)

```
F={Num → Date ; Num → Client ;
 Article → Prix ; Article, Num→ Quantité}
```



II.6. LA FERMETURE TRANSITIVE

Si F est une *couverture fonctionnelle* de la relation $\mathcal{R}$, *la fermeture transitive* de F *notée F+* est l'ensemble des dépendances fonctionnelles qui peuvent être inférées de F par transitivité. ⟹ On appelle fermeture transitive F+ d'un ensemble F de DFE*, l'ensemble de toutes les DFE qui peuvent être composées par transitivité à partir des DFE de F.

Exemples :

1. F={`Châssis → Type; Type → Marque; Type → Puissance`}

   F+=F ⋃ `{Châssis→ Marque; Châssis →Puissance}`

***Note** : Deux ensembles de dépendances fonctionnelles sont équivalents s'ils ont la même fermeture transitive.*

II.7. LA COUVERTURE MINIMALE

Une couverture minimale appelée aussi couverture *irrédondante* et notée CM(F) est l'ensemble F de dépendances fonctionnelles élémentaires associé à un ensemble d'attributs vérifiant les propriétés suivantes :

- ✓ Aucune dépendance dans F n'est redondante ;
- ✓ Toute dépendance fonctionnelle élémentaire des attributs est dans F+;
- ✓ C'est le sous-ensemble minimal de dépendances fonctionnelles permettant de générer toutes les autres.
  1- $\forall X \rightarrow Y \in F'$, Y ne comporte qu'un seul attribut;
  2- $\forall X \rightarrow A \in F'$, l'ensemble $F'-\{X \rightarrow A\}$ n'est pas équivalent à F';
  3- $\forall X \rightarrow A \in F'$ et $Z \subset X$, l'ensemble $F'-\{X \rightarrow A\} \cup \{Z \rightarrow A\}$ n'est pas équivalent à F'

Ainsi, la couverture minimale d'un ensemble de DFs est un sous-ensemble minimum de DFE non transitives permettant de générer toutes les autres au moyen des propriétés d'Armstrong.

***Note** : Tout ensemble de DFE (et donc tout ensemble de DF) admet au moins une couverture minimale (et en pratique souvent plusieurs).* ⇨*Il n'y a pas unicité de la couverture, elle dépend de l'ordre choisi*.

ALGORITHME DE CALCUL DE LA CM D'UN ENSEMBLE DE DFs

- **Entrée :** $R=(A_1,A_2,...,A_n)$ un schéma de relation; F la couverture fonctionnelle de R;
- **Sortie :** G une couverture minimale de F, définie par
  1- G+ = F+;
  2- Tout membre droit d'une DF de G est réduit à un seul attribut;
  3- Pour aucune DF $X \rightarrow A \in G$ on n'a $G-\{X \rightarrow A\} \equiv X \rightarrow A$;
  4- Pour aucune DF $X \rightarrow A \in G$ on n'a $G \equiv Y \rightarrow A$ avec $Y \subset X$;

Algorithme

On ordonne les DFs de F, supposons :$F=\{X_1 \rightarrow Y_1, X_2 \rightarrow Y_2, ..., X_m \rightarrow Y_m\}$

Etape 1 : on décompose les membres droits Y_i des DFs de F

Pour i←1 à m faire
  Si $Y_i=A_1A_2...A_s$, avec s>1
    alors $F \leftarrow F-\{X_i \rightarrow Y_i\} \cup \{X_i \rightarrow A_1, X_i \rightarrow A_2, ..., X_i \rightarrow A_s\}$

Nous supposons par la suite que :$F=\{X_1 \rightarrow A_1, X_2 \rightarrow A2, ..., X_p \rightarrow A_p\}$

Etape 2 : on regarde si on peut enlever des dfs de F sans modifier sa fermeture

Pour i←1 à p faire
si $F-\{X_i \rightarrow A_i\} \equiv \{X_i \rightarrow A_i\}$ alors $F \leftarrow F-\{X_i \rightarrow A_i\}$

Quitte à renuméroter les DFs, nous supposons qu'à la fin de cette étape nous avons : $F=\{X_1 \rightarrow A_2, X_2 \rightarrow A_2, ..., X_q \rightarrow A_q\}$ avec $q \leq p$.

Etape 3 : on cherche si on peut remplacer les membres gauches des DFs de F qui sont formés de plus d'un attribut par des membres gauches comprenant moins d'attributs sans changer la fermeture de F.

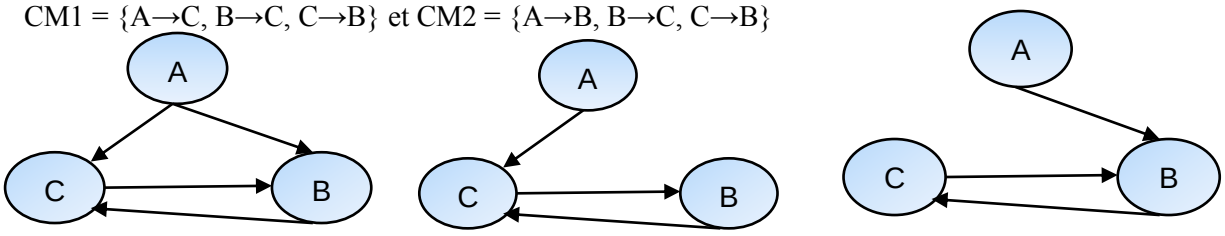
Pour i←1 à q faire
 Si $Xi=B_1B_2...B_r$, avec r>1
   Alors pour j←1 à r faire
     Si $F \equiv \{X_i-B_j\} \rightarrow A_i$, alors $X_i \leftarrow \{X_i-B_j\}$

Il n'y a pas unicité de la couverture, elle dépend de l'ordre choisi.

Exemple 1 :

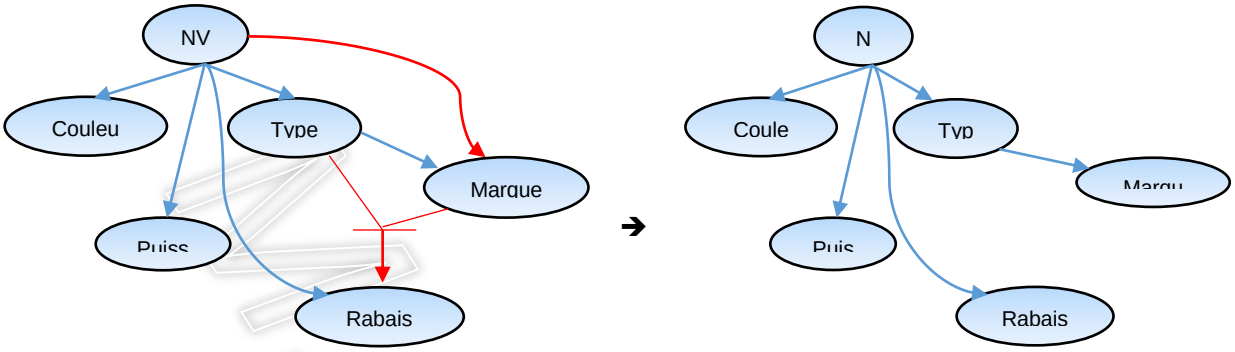
L'ensemble F = {A→B, A→C, B→C, C→B} admet les deux couvertures minimales :

CM1 = {A→C, B→C, C→B} et CM2 = {A→B, B→C, C→B}



Exemple 2 :

- D = { NV → Type ; Type → Marque ; NV → Couleur ; NV → Marque ; NV → Puiss ; (Type, Marque) → Rabais ; NV → Rabais}
- Couverture minimale :
  CM(D) = { NV → Type ; Type → Marque ; NV → Couleur ; NV → Puiss ; NV → Rabais}



III. NORMALISATION DES RELATIONS (THEORIE DE LA NORMALISATION)

Cette théorie est basée sur les dépendances fonctionnelles qui permettent de *décomposer* l'ensemble des informations en diverses relations sans *perte d'informations*. Chaque nouvelle forme normale marque une étape supplémentaire de la progression vers des *relations présentant de moins en moins de redondance*.

III.1. POURQUOI NORMALISER ?

- ✓ Pour limiter les redondances de données,
- ✓ Pour limiter les pertes de données,
- ✓ Pour limiter les incohérences au sein des données,
- ✓ Pour améliorer les performances des traitements.

III.2. LES FORMES NORMALES

III.2.1. PREMIERE FORME NORMALE (1NF)

Une relation est *en première forme normale* si et seulement si tous ses attributs ont des valeurs simples (non multiples, non composées). ⟹ Une relation est en 1NF si et seulement si tous ses attributs *sont atomiques*. Un attribut est atomique s'il ne contient qu'une seule valeur pour un *tuple* donné, et donc s'il ne regroupe pas un ensemble de plusieurs valeurs.

Exemple

`ETUDIANT(NUM,NOM,PRENOM)` `LIVRE (CODE, TITRE, AUTEUR)`

ETUDIANT

NUM	NOM	PRENOM
1	Dupont	Pierre Jean
2	Durand	Marie
3	Dupré	Sylvie Claudine Claire

LIVRE

CODE	TITRE	AUTEUR
100	L'art des BD	Miranda Busta

La valeur de l'attribut PRENOM n'est pas simple.

Normaliser en première forme normale, il existe deux solutions :

- **1ère solution :** Créer autant d'attributs que le nombre maximum de valeurs de l'attribut multi-valué (*stockage horizontal*). On utilise cette solution quand le nombre maximum de valeurs est connu à l'avance et qu'il ne risque pas de changer plus tard. Par exemple pour représenter les parents d'une personne, on sait qu'une personne ne peut avoir plus de deux parents donc on décompose en deux attributs un pour le père et un pour la mère.

ETUDIANT

NUM	NOM	PRENOM1	PRENOM2	PRENOM3
1	Dupont	Pierre	Jean	*NULL*
2	Durand	Marie	*NULL*	*NULL*
3	Dupré	Sylvie	Claudine	Claire

- **2ème solution :** Créer une nouvelle relation comportant la clef de la relation initiale et l'attribut multi-valué puis éliminer l'attribut multi-valué de la relation initiale (*stockage vertical*). Cette solution est utilisée quand on ne connait pas le nombre maximum de valeur ou si ce dernier et trop important.

LIVRE	CODE	TITRE
	100	L'art des BD

AUTEURS	CODE	AUTEUR
	100	Miranda
	100	Busta

Remarque : *on vient de faire apparaître un couple clef primaire/clef étrangère*

INCONVENIENTS
- **Solution 1** : stockage de valeurs nulles, impossibilité de stocker plus de valeur que prévu
- **Solution 2** : opération de jointure, lourdeur des auto-jointures

AVANTAGES
- **Solution 1 :** tout est dans la même relation (pas de jointure)
- **Solution 2** : pas de valeur nulle, pas de limite au stockage

CHOIX
- **Solution 1 :** quand le nombre de valeurs de l'attributs multi-valué est constant ou que le nombre max de valeur est faible (très peu de valeurs nulles)
- **Solution 2 :** dans les autres cas

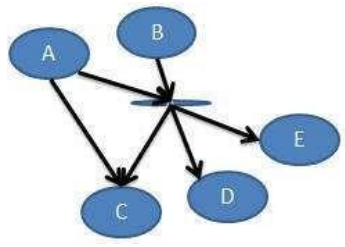
III.2.2. DEUXIEME FORME NORMALE (2NF)

Une relation est en deuxième forme normale si et seulement si :
- ✓ Elle est 1FN et
- ✓ Tous les attributs "non clés" sont *complètement dépendants* de la clé primaire (mais peuvent l'être transitivement). i.e. ils ne peuvent pas être dépendant d'une partie de la clé. Toute la clé est nécessaire pour retrouver l'attribut "non clé".

Remarque :
La **2NF** est basée sur le concept de dépendance fonctionnelle *complète*. Une dépendance fonctionnelle X → Y est complète si l'élimination d'un attribut x_i quelconque de X détruit la dépendance fonctionnelle.

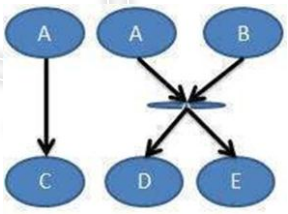
Exemples :

1. Soit :R(A, B, C, D, E)
F={AB→C, D, E ; A→C} ➔ la clé={A,B}
La relation suivante n'est pas en deuxième forme normale.



Normaliser en deuxième forme normale

Pour normaliser une relation en deuxième forme normale, il faut :
- ✓ Isoler la DF *partielle* dans une nouvelle relation ;
- ✓ Enlever la cible de cette DF de la relation initiale ;



2. `Enseignement (NUM, CODE_MATIERE, NOM, VOLUME_HORAIRE)`
`Avec : NUM→Nom`

Normalisation de l'exemple ci-dessus
`Enseignement (NUM, CODE_MATIERE, VOLUME_HORAIRE)`
`Enseignant (NUM, NOM)`

Note : Une relation en 1NF dont la clé est mono-attribut (se compose d'un seul attribut) est forcément en 2NF.

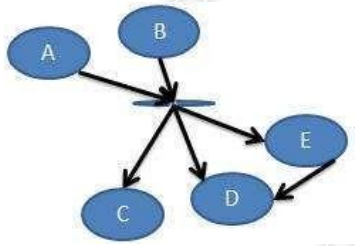
III.2.3. TROISIEME FORME NORMALE (3NF)

Une relation est en troisième forme normale si et seulement si :

- ✓ Elle est en deuxième forme normale et
- ✓ Tout attribut n'appartenant pas à une clé ne dépend pas d'un autre attribut non clé. ⟹ tous les attributs " non-clés " sont dépendants non transitivement (*dépendants directement*) de la clé primaire.

Exemples :

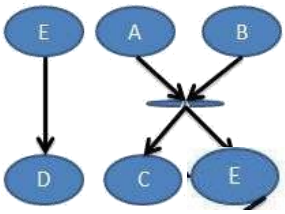
1. Soit :R(A, B, C, D, E)
F={AB→C, D, E ; E→D}➔ la clé={A,B}
La relation suivante n'est pas en troisième forme normale.



Normaliser en troisième forme normale

Pour normaliser une relation en troisième forme normale, il faut :

- ✓ Isoler la DF *transitive* dans une nouvelle relation ;
- ✓ Eliminer la cible de cette DF de la relation initiale ;



2. ENSEIGNANT (**NUM**, NOM, CATÉGORIE, CLASSE, SALAIRE)
Avec CATÉGORIE, CLASSE → SALAIRE

Normalisation de l'exemple ci-dessus

ENSEIGNEMENT (NUM, NOM, CATÉGORIE, CLASSE)
SALAIRE (**CATÉGORIE, CLASSE**, SALAIRE)

Note : La 3NF exprime le fait que tous les attributs non clé dépendent complètement et uniquement (directement) de la clé de la relation.

Il est souhaitable que les relations logiques soient en 3NF. En effet, il existe toujours une décomposition sans perte d'information et préservant les DF d'un schéma en 3NF.

III.2.4. FORME NORMALE DE BOYCE CODD(BCNF)

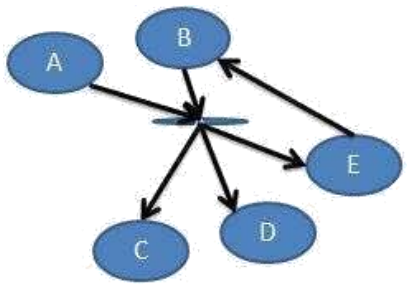
Une relation est en forme normale de Boyce Codd si et seulement si :

- ✓ Elle est en troisième forme normale et
- ✓ Toute source de dépendance fonctionnelle est une clé primaire minimale. ⟹Les seules DFE existantes sont celles pour lesquelles une clé candidate détermine un attribut.

Exemples :

1.

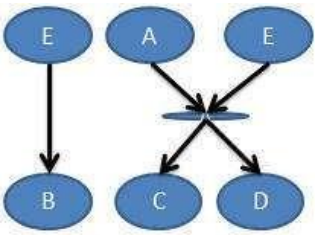
Soit :R(A, B, C, D, E)
F={AB→C, D, E ; E→B}➔ la clé={A,B}
La relation suivante n'est pas BCFN.



Normaliser en BCFN

Pour normaliser une relation en troisième forme normale, il faut :

- ✓ Isoler la DF problématique dans une nouvelle relation ;
- ✓ Éliminer la cible de cette DF et la remplacer par sa source dans la relation initiale.



2.

Soit la relation Personne :

```
1 Personne (N°SS, Pays, Nom, Région)
Soit les DF suivantes sur cette relation :
{N°SS,Pays→Nom ; N°SS,Pays→Région ; Région→Pays}
```

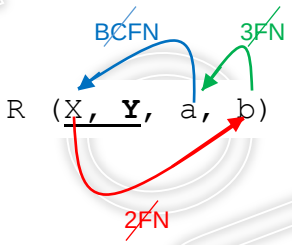
Il existe une DFE qui n'est pas issue d'une clé et qui détermine un attribut appartenant à une clé. Cette relation est en 3NF, mais pas en BCNF (car en BCNF toutes les DFE sont issues d'une clé).
Pour avoir un schéma relationnel en BCNF, il faut décomposer Personne :

```
1 Personne (N°SS, Région, Nom)
2 Région (Région, Pays)
```

Remarquons que les DF n'ont pas été préservées par la décomposition puisque N°SS et Pays ne déterminent plus Région.

Attention *: Une décomposition en BCNF ne préserve pas toujours les DF.*

III.3. SYNTHESE



IV. ELABORATION D'UNE BASE RELATIONNELLE

Elaboration mal adroite d'une base de données expose aux anomalies suivantes :

- Redondance (répétition) de données
- Inaptitude à la représentation de certaines informations.
- Perte d'informations.

IV.1. REPETITION DE DONNEES

Soit la relation qui gère les prêts des clients caractérisés par leur montant, d'une agence bancaire caractérisée par son avoir et sa ville.
Emprunt (Agence, Avoir, Ville, Prêt, client, montant)

Agence	Avoir	Ville	Prêt	Client	Montant
A_1	V_1	T	P_1	C_1	M_1
A_1	V_1	T	P_2	C_2	M_2
A_2	V_2	T	P_3	C_3	M_3
A_3	V_3	T	P_4	C_4	M_4

Anomalies :

- Si on veut ajouter un prêt dans une agence, il faut répéter les attributs agence, avoir, ville.
- Si par exemple une agence A1 change de ville, il faut modifier tous les tuples où apparait l'agence A1.

IV.2 L'INAPTITUDE A LA REPRESENTATION DES INFORMATIONS

Soit la relation qui gère les montants des comptes clients :
Mouvement (Agence, Prêt, montant, compte, client)

Anomalie :
Si un client possède un compte dans l'agence sans avoir un prêt alors il aura un problème de représentation du tuple associé à ce client, puisque l'attribut prêt est sans valeur.

IV.3. PERTE D'INFORMATIONS

Avant d'expliquer ce concept, il faut d'abord introduire deux opérateurs de l'algèbre relationnelle ; la *projection* et *jointure naturelle*.

- **PROJECTION**

  La projection d'une relation $\mathcal{R}(A_1, A_2,\ldots A_n)$ sur les attribut $A_1, A_2,\ldots A_p$ est une relation $\mathcal{R}'(A_1, A_2\ldots\ldots, A_p)$ obtenu par élimination des valeurs des attributs de $\mathcal{R}$ n'appartenant pas à $\mathcal{R}'$ et suppression des tuples en double. La notation $\mathcal{R}'(A_1, A_2\ldots\ldots, A_p)$ = Proj ($\mathcal{R}$ /A_1, $A_2\ldots\ldots, A_p$)

  Exemple : $\mathcal{R}$(A, B, C, D) **$\mathcal{R}_1$=Proj($\mathcal{R}$/A, C)**

A	B	C	D
A_1	B_1	C_1	D_1
A_2	B_2	C_2	D_2
A_1	B_3	C_1	D_1
A_3	B_2	C_1	D_2

→

A	C
A_1	C_1
A_2	C_2
A_1	C_1
A_3	C_1

→

A	C
A_1	C_1
A_2	C_2
A_3	C_1

- **LA JOINTURE NATURELLE**

  Soient deux relations $\mathcal{R}(A_1, A_2, \ldots A_n)$ et $\mathcal{S}(B_1, B_2,\ldots.B_k)$, Join ($\mathcal{R}$, $\mathcal{S}$)= $T(X_1, X_2,\ldots. X_m)$ tel que : $\{X_1, X_2,\ldots. X_m\}=\{A_i\}.\{B_i\}$⇨ les tuples de T sont obtenus par concaténation des tuples de $\mathcal{R}$ et $\mathcal{S}$ qui ont la même valeur pour les attributs communs.

  Exemple :

  Soient les deux relations :

  - Client (N° client, montant,)
  - Dossier (agence, prêt, montant)

Client

N° client	Montant
C_1	M_1
C_2	M_2
C_3	M_3
C_4	M_1
C_5	M_4

Dossier

Agence	Prêt	Montant
A_1	P_1	M_1
A_1	P_2	M_2
A_2	P_3	M_3
A_3	P_4	M_1
A_3	P_5	M_4

R= Join(Client, Dossier)

N° client	Montant	Agence	Prêt
C_1	M_1	A_1	P_1
C_1	M_1	A_3	P_4
C_2	M_2	A_1	P_2
C_3	M_3	A_2	P_3
C_4	M_1	A_1	P_1
C_4	M_1	A_3	P_4
C_5	M_4	A_3	P_5

On dit qu'il y pas de perte d'information lorsque le résultat de la jointure naturelle entre deux relations R_1, R_2 ayant comme attributs des sous attributs de R est égale à l'ensemble des tuples de la relation de départ R.

- **DECOMPOSITION DE RELATIONS**
  - **Principe :** remplacement **d'une relation** $\mathcal{R}(A_1, A_2, \ldots A_n)$ par une collection de relations $\mathcal{R}_1$, $\mathcal{R}_2$……$\mathcal{R}_k$ obtenus par des productions de R tel que la relation résultats des jointures des $\mathcal{R}_i$ avec i ∈ {1..k} ait le même schéma relationnel.
  - **Qualité :** une décomposition doit être :
    - **Sans perte d'information : i.e.** que la jointure naturelle doit être conservée.
    - **Préservation des DFs :** soit R une relation et F sa couverture fonctionnelle et soit une décomposition $\mathcal{R}_1$, $\mathcal{R}_2$,…….$\mathcal{R}_n$ de $\mathcal{R}$, cette décomposition préserve les DFs sis et seulement si la fermeture transitive F+ de F est identique à celle de l'union des fermeture transitives F_i^+ des R_i avec i ∈{1…n}

Remarque : *il a été prouvé que toute relation à une décomposition en 3FN est sans perte d'information et qui préserve les DFs.*

Plusieurs approche permettent cette décomposition nous présentons une approche basée sur l'algorithme de *BERNSTEIN*.

L'algorithme de Synthèse 'BERNSTEIN '

Cet algorithme est basé sur la décomposition de la relation universelle, la relation universelle = relation définie sur tous les attributs.

- **Entrée** : $\mathcal{R}$={U, F} U= ensemble de tous les attributs , F= l'ensemble de DFs.
- **Sortie** : S={R_1, R_2,…..R_n} tel que R_i={X_i, F_i'} et R_i en 3FN ∀ i ∈{1...n}

  - **Etape1 :** Pour chaque DF « *f* » rendre *f élémentaire* soit F' cet ensemble.
  - **Etape 2 :** Rechercher une couverture minimale de F' notée *min (F')*. On cherche les clés minimales de R et on teste si R est en 3FN. Si oui, on s'arrête. Sinon :
  - **Etape 3 :** Partitionner *min (F')* en groupes F_1', F_2'……F_k' tel que les DFs d'un groupe aient la même partie gauche.
  - **Etape 4 :** Pour chaque groupe F_i' construire un schéma R_i=> <X_i, F_i'>où X_i est l'ensemble des attributs apparaissant dans F_i'
  - **Etape 5 :** Si aucun des schémas R_i définis à l'étape 4 ne contient de clé de R, on ajoute un schéma R_K=(K), où K est une clé de R, muni d'aucune DF.

Exemple :

Considérons le schéma R=(C, E, P, N, J, H, S) muni de sa couverture fonctionnelle :
F={C, E, P→N (1) ; J, H, S→PC (2) ; E, P→C (3) ; E, P, S→C (4)}

Etape 1 :

On a:

- C, E, P→N (1) ; n'est pas élémentaire car : {E, P→C (3), CEP→N (1)} ≡ EP→N (1') *par pseudo-transitivité*;
- J,H,S→P(2') ;
- J,H,S→C (2'') ;
- E,P→C (3) ;
- E, P, S→C (4) n'est pas élémentaire car on a E, P→C (3) ;

Etape 2 :

On remplace F par sa couverture minimale, à savoir : F'= {EP→N (1') ; JHS→P (2') ; JHS→C (2'') ; EP→C (3)}. Toute clé minimale de R doit contenir les attributs EJHS qui ne figurent pas dans les membres droits des DF de F'. On vérifie facilement que EJHS est la seule clé de R et R n'est pas en 3FN.

Etape 3 :

On regroupe les DFs (1') et (3), puis (2') et (2''), et on définit : F'_1={(1'), (3)}, F'_2={(2'), (2'')}

Etape 4 :

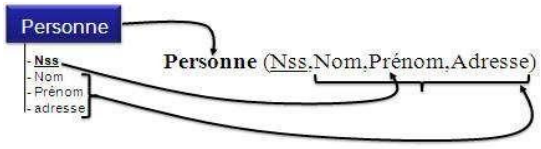
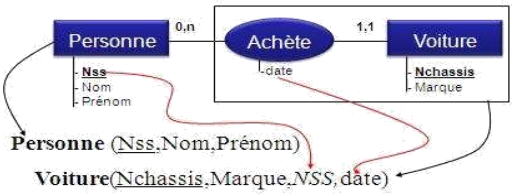
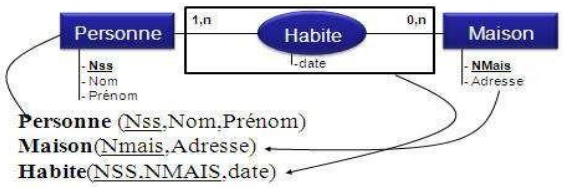
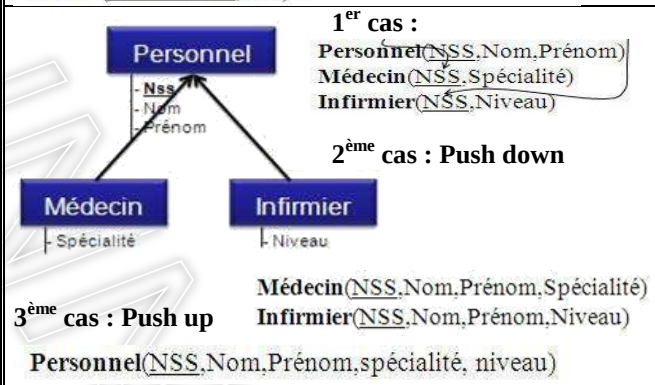
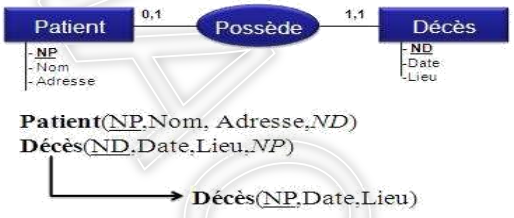
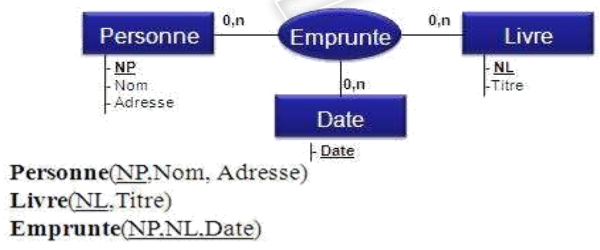
- R_1=< (**J, H, S**, C, P), F'_1 = {J, H, S→ P; J, H, S→C}>
- R_2=< (**E, P**, C, N), F'_2={E, P→ C ; E, P→N}>

Etape 5 :

Ni R_1, ni R_2 ne contiennent EJHS la clé de la relation de départ R, par suite on ajoute le schéma : R_3=(E, J, H, S) muni d'aucune DF.

V. PASSAGE DE L'ENTITE-ASSOCIATION AU MODELE RELATIONNEL

Le modèle Entité/Association est un modèle conceptuel permettant de modéliser une réalité à l'aide de trois concepts essentiels : l'entité, l'association et les propriétés. Au niveau du relationnel on ne traite que des *relations*. Pour passer du premier modèle au second on a besoin de règles pour faire la correspondance entre les concepts de l'un et de l'autre.

Règle 1 : TYPE ENTITE NON FAIBLE Un type-entité E non faible est représenté par une relation T dont les propriétés simples sont les attributs de l'entité E et la clé de T est l'identifiant de E.	
Règle 2 : RELATION 1:N Dans le cas d'une relation 1: n (Père/fils), l'association n'est pas représentée par une relation, cependant ses propriétés migrent vers la relation représentant le fils et l'identifiant du père migre vers le fils comme *clé étrangère.*	
Règle 3 : RELATION N:M Dans le cas d'une relation n:m, *l'association A* est représentée par une *relation T* dont les propriétés sont les propriétés de A et la clé est la concaténation des identifiants des entités participant à l'association A.	
Règle 4 : GENERALISATION/ SPECIALISATION Dans le cas d'une généralisation / spécialisation, il existe trois manières de passer au schéma relationnel :	
ASSOCIATION 1:1 Dans le cas d'une association 1:1, la migration de l'identifiant peut se faire dans un *seul sens* comme on peut le faire dans les *deux sens*. Il est à noter que quand un cas pareil se présente il est presque impossible que les *cardinalités minimales des deux côtés soient toutes les deux égales à 1*, l'une d'elles est toujours un **0.** Ce qui fait que le sens le plus recommandé pour la migration de l'identifiant est celui allant du côté de la cardinalité minimale **0** vers celui de la cardinalité minimale **1**.	
CAS PARTICULIER : ENTITE AVEC UN SEUL ATTRIBUT Quand on est en présence d'une entité ne possédant qu'un seul attribut qui est son identifiant dans le modèle Entité Association alors cette entité n'est pas représentée dans le schéma relationnel correspondant.	

VI. RESUME

- ✓ La première forme normale exprime le fait que toutes les valeurs des attributs *sont atomiques* (simples, non composées).
- ✓ La seconde forme normale exprime le fait que tous les attributs non clé d'une relation *dépendent complètement* de la clé de cette dernière et non pas d'une seule partie de cette clé.
- ✓ La troisième forme normale ajoute une autre restriction à la seconde forme normale en exprimant le fait que tous les attributs non clé *dépendent complètement et uniquement* de la clé de la relation.
- ✓ La Boyce Codd forme normale pousse plus loin la restriction de la troisième forme normale en exprimant le fait que dans toute relation en troisième forme normale, s'il existe une dépendance fonctionnelle alors sa partie gauche est forcément une clé de cette relation.
- ✓ Le passage d'une forme normale à une autre est exprimé par le schéma ci-dessous.