CHAPIPTRE 2

Conception d'une base de données

Le modèle relationnel s'intéresse à trois aspects des données : leur **structure, leur intégrité et leur manipulation.** Nous nous intéressons dans ce chapitre au premier aspect. Lors d'une conception d'une base de données relationnelle la première étape consiste à définir le schéma relationnel. Il existe plusieurs possibilités de schémas, nous sommes amenés à choisir les meilleurs. Encore faut-il définir les critères qui les qualifient et concevoir les algorithmes qui permettent de les obtenir. Codd, le premier proposa en 1971 les bases d'une démarche de normalisation qui conduit à décomposer par projection une relation selon plusieurs schémas plus restreints, normalisés, sans perte d'informations. Cette démarche repose aujourd'hui sur un ensemble rigoureux de règles et la mise en œuvre d'algorithmes performants. Avant de présenter cette démarche quelques définitions de relations, de clés, de dépendance fonctionnelle, etc. ... sont nécessaires

3.1. Notions de bases.

Définition 1. Relation : *Soient D1, D2, …,, Dn des ensembles de valeurs (domaines) pas nécessairement distincts. Une relation R (D1, D2, …, Dn) est un sous ensemble du produit cartésien des domaines D1, D2, … Dn, i.e., R (D1, D2, … Dn)* $\subset$ *D1 X D2 X … X Dn.*

Un tuple ou n-uplet d'une relation est < d1, d2, …,dn> : d1 $\in$ D1, … , dn $\in$ Dn.

En plus de la prise en charge des règles spécifiques à une base de données, le modèle relationnel inclut deux propriétés d'intégrité générales :

1. les clés candidates et primaires
2. les clés étrangères

Nous avons déjà fait référence de façon informelle aux clés primaires : une clé primaire d'une relation est simplement un identificateur unique de cette relation. Il faut savoir qu'une clé primaire est un cas particulier du concept plus fondamental de clé candidate.

Définition 2. Clé candidate : *Soit R une relation, une clé candidate C pour cette relation est un sous ensemble de l'ensemble des attributs de R tel que :*

1. *Il n'existe pas deux n-uplets distincts de toute l'extension courante de R ayant la même valeur C* ***(propriété d'unicité)***
2. *Aucun sous-ensemble strict de C ne possède la propriété d'unicité* ***(propriété d'irréductibilité).***

Remarques :

- Chaque relation possède, au moins, une clé candidate parce que les relations ne contiennent pas de n-uplets dupliqués. Il s'ensuit qu'au moins la combinaison de tous les attributs de la relation possède la propriété d'unicité, de là, soit cette combinaison possède la propriété d'irréductibilité et est donc une clé candidate (en fait la seule clé candidate), soit il existe au moins un sous-ensemble strict de cette combinaison qui a la propriété d'unicité et également la propriété d'irréductibilité

- Si la relation a plus d'une clé candidate, le modèle relationnel exige que seule une de ces clés candidates soit choisie comme clé primaire pour cette relation ; les autres clés, s'il y en a, sont appelées ***clés alternatives.***

Définition 3. Clé étrangère : *Soit R2 une relation de la base, alors une clé étrangère EC dans R2 est un sous ensemble des attributs de R2 tel que*

1. *Il existe une relation de base R1 (R1 et R2 ne sont pas nécessairement distinctes) avec une clé C et,*
2. *à tout instant, chaque valeur EC dans l'extension courante de R2 est identique à la valeur C dans un n-uplet particulier de l'extension courante de R1.*

Exemple :

Soit la base de données suivante décrivant les départements à l'aide de la relation DEPT et les employés travaillant dans ces départements à l'aide de la relation EMP :

DEPT(NDEPT, DNOM, BUDGET)
EMP(NEMP, ENOM, *NDEPT*, SAL)

Dans cette base de données, l'attribut NDEPT de la relation EMP est une clé étrangère, qui s'unifie avec la clé primaire NDEPT de la relation DEPT. Cependant, ce n'est pas un composant de la clé primaire NEMP, ni d'aucune autre clé candidate de la relation EMP.

3.2 Concept de dépendance fonctionnelle

Afin de dégager quelques propriétés d'un schéma relationnel mal conçu, considérons le schéma modifié de notre exemple sur les fournitures dans lequel on rajoute l'adresse du fournisseur :

VFOURNITURE (NF, VILLE, NP, QTE) dont l'extension est la suivante :

NF	VILLE	NP	QTE
F1	Alger	P1	100
F1	Alger	P2	100
F1	Alger	P3	100
F1	Alger	P4	100
F1	Alger	P5	100
F1	Alger	P6	100
F2	Tunis	P1	200
F2	Tunis	P2	200
F3	Tunis	P2	300
F4	Alger	P2	400
F4	Alger	P4	400
F4	Alger	P5	400

Cet exemple soulève plusieurs types de problèmes :

1. Redondance : l'adresse d'un fournisseur est répétée pour chaque pièce qu'il fournit

2. Anomalie de mise à jour : lors d'une mise à jour, on risque de modifier l'adresse d'un fournisseur dans un n-uplet mais pas dans les autres. Un même fournisseur possèdera plusieurs adresses !
3. Anomalies d'insertion : on n'enregistre pas l'adresse d'un fournisseur, si on n'achète pas au moins une pièce à ce fournisseur
4. Anomalies de suppression : si on supprime toutes les pièces proposées par un fournisseur ce dernier n'apparaît plus.

Le schéma précédent se décompose selon deux schémas :

FV (NF, VILLE)
FOURNITURE (NF, NP, QTE)

Cette décomposition élimine les problèmes précédents. En revanche, pour trouver l'adresse d'un fournisseur qui fournit une pièce donnée, il faut recourir à une jointure entre FV, et FOURNITURE. L'opération peut se révéler coûteuse en temps, mais si on doit choisir entre la cohérence de la base de données et le gain de temps c'est toujours la cohérence qui prime.
Les procédures qui permettent de décomposer une relation en ″ bonnes relations″, reposent sur le concept de dépendance fonctionnelle (DF).

Définition 4 Dépendance fonctionnelle : *Soit R une relation, soient X et Y deux sous-ensembles quelconques de l'ensemble des attributs de R. On dit que X détermine Y ou que Y est en dépendance fonctionnelle avec X noté X* $\rightarrow$ *Y si et seulement si, pour n'importe quelle extension de R, à chaque valeur de X dans R correspond exactement une seule valeur de Y dans R.*

Le membre gauche d'une dépendance fonctionnelle est appelée déterminant et le membre droit dépendant

Voici quelques dépendances fonctionnelles qui s'appliquent à la relation VFOURNITURE :

{NF, NP} $\rightarrow$ QTE
{NF, NP} $\rightarrow$ VILLE
{NF, NP} $\rightarrow$ {VILLE, QTE}
{NF, NP} $\rightarrow$ NF
{NF, NP} $\rightarrow$ {NF, NP, VILLE, QTE}
{NF, NP} $\rightarrow$ NP

Les dépendances fonctionnelles : NF $\rightarrow$ QTE et QTE $\rightarrow$ NF existent dans VFOURNITURE, mais ne sont pas valides à tout instant. En effet, l'énoncé ″pour un fournisseur donné, toute cargaison a la même quantité de pièces livrées″, est vrai pour l'exemple mais n'est pas vrai tout le temps.

Remarque : Si X est une clé candidate de la relation R, en particulier, s'il s'agit de la clé primaire, alors tous les attributs Y de la relation R doivent nécessairement être en dépendance fonctionnelle avec X (ce fait est une conséquence de la définition de la clé candidate).

3.2.1 Dépendances triviales

Une dépendance fonctionnelle est une contrainte d'intégrité que le SGBD doit à chaque fois vérifier lors des mises à jour. Le but est donc de réduire au maximum le nombre de celles-ci. Parmi les dépendances que nous avons trouvées, certaines sont dites triviales que nous pouvons éliminer.

Définition 5 Dépendance triviale : *Une dépendance fonctionnelle est dite triviale si et seulement si le membre droit est un sous-ensemble (pas nécessairement strict) du membre gauche.*

Nous avons par exemple trouvé :

$$\{NF, NP\} \rightarrow NF$$

3.2.2. Fermeture des dépendances fonctionnelles :

L'ensemble de toutes les dépendances fonctionnelles qui sont impliquées par un ensemble donné *F* de dépendances fonctionnelles est appelé fermeture de *F* dénoté *F+*.

Les axiomes ci-dessous, appelés règles d'inférences d'Armstrong sont un moyen de calculer cette fermeture. Soient A, B, C trois sous-ensembles d'attributs de l'ensemble des attributs de la relation R, et admettons (par exemple) qu'AB représente l'union de A et B. Alors :

1. **Réflexivité** : Si B est un sous-ensemble de A , alors $A \rightarrow B$
2. **Augmentation** : Si $A \rightarrow B$, alors $AC \rightarrow BC$
3. **Transitivité** : Si $A \rightarrow B$ et $B \rightarrow C$, alors $A \rightarrow C$

Plusieurs règles supplémentaires peuvent être dérivées des trois règles représentées ci-dessus. Ces règles peuvent être utilisées pour simplifier le calcul de F+ à partir de F. Si D est un autre sous-ensemble quelconque de l'ensemble des attributs de R

5. **Pseudo - Transitivité**: Si $A \rightarrow B$ et $CB \rightarrow D$ alors $CA \rightarrow D$
6. **Décomposition** : Si $A \rightarrow BC$, alors $A \rightarrow B$ et $A \rightarrow C$
7. **Union** : Si $A \rightarrow B$ et $A \rightarrow C$ alors $A \rightarrow BC$
8. **Composition** : Si $A \rightarrow B$ et $C \rightarrow D$ alors $AC \rightarrow BD$

Exemple : R(A,B ,C,D,E,F)

$F = \{ A \rightarrow BC,\ B \rightarrow E,\ CD \rightarrow EF \}$

On peut montrer que $AD \rightarrow F$ est un élément de la fermeture de cet ensemble.

1. $A \rightarrow BC$ (donnée)
2. $A \rightarrow C$ (1, décomposition)
3. $AD \rightarrow CD$ (2, augmentation)
4. $CD \rightarrow EF$ (donnée)
5. $AD \rightarrow EF$ (3, 4, transitivité)
6. $AD \rightarrow F$ (5, décomposition)

$AD \rightarrow F$ est bien un élément de la fermeture de cet ensemble

3.2.3. Fermeture d'un ensemble d'attributs

En général, la génération directe de la fermeture transitive *F+*, relative à un ensemble de dépendances fonctionnelles *F* se révèle coûteuse en temps, car le cardinal de *F+* peut être très grand même si *F* est restreint. Par exemple, si

$F = \{ A \rightarrow B1, A \rightarrow B2, \ldots, A \rightarrow Bn \}$

F+ inclut toutes les dépendances de type :

$A \rightarrow Y$ où Y est un sous-ensemble de ce type. On ne peut espérer générer *F+* en un temps raisonnable même pour n relativement petit. Par contre il y a un moyen efficace pour déterminer si une DF donnée appartient à cette fermeture. Pour cela, nous avons besoin d'introduire la notion de **super-clé :**

Définition 6. Super-clé. *Une super-clé pour une relation R est un ensemble d'attribut de R qui inclut au moins une clé candidate de R en tant que sous-ensemble. Ce sous-ensemble n'est pas nécessairement strict (la définition de super-clé peut ainsi être dérivée de celle de clé candidate en supprimant simplement la contrainte d'irréductibilité).*

Il s'ensuit que les super-clés, pour une relation donnée R sont précisément les sous-ensembles de *C* de l'ensemble des attributs de R, tels que la dépendance fonctionnelle $C \rightarrow A$ est valide pour chaque attribut A de R.

Les clés candidates sont les super-clés qui sont irréductibles. Donc, si nous connaissons les dépendances fonctionnelles d'une relation et nous voulons déterminer les clés candidates, il suffit de considérer les super-clés qui sont irréductibles.

Pour déterminer si *C* est une super-clé , nous avons besoin de déterminer si l'ensemble de tous les attributs dépendants fonctionnellement de *C* est en fait l'ensemble de tous les attributs de R , c'est à dire , calculer la fermeture de *C* dénotée *C+*.

L'algorithme général suivant permet de calculer cette fermeture :

```
Début
fermeture  [C, F] : = C
faire  ″pour toujours″ ;
        pour chaque DF  X  →  Y   de  F
                faire :
                        Si X est un sous-ensemble de fermeture [C, F]
                        alors fermeture [C, F] : = fermeture [C, F] ∪ Y;

                fin ;
        Si fermeture [C, F] n'est pas modifiée pendant cette itération, alors
        /* calcul terminé  */ quitter  la boucle ;
Fin ;
```

Exemple :

Supposons que soit donnée une relation R (A, B, C, D, E, F) et les dépendances fonctionnelles {A $\rightarrow$ BC, E $\rightarrow$ CF, B $\rightarrow$ E, CD $\rightarrow$ EF}
Nous pouvons calculer {A, B}$^+$ de l'ensemble des attribut {A,B} pour cet ensemble de dépendances fonctionnelles.

1. initialisation de *fermeture* [C, F] à {A, B}
2. répétons la boucle interne quatre fois, une fois par DF. Lors de la première itération A $\rightarrow$ BC, le membre gauche (A) appartient à *fermeture (C, F)* donc nous lui ajoutons les attributs B et C au résultat. D'où *fermeture[C, F]* = {A, B, C}
3. à la troisième itération B $\rightarrow$ E, nous ajoutons E à *fermeture[C, F]* qui est égale à {A, B, C, E}
4. à la quatrième itération CD $\rightarrow$EF, *fermeture [C, F]* reste inchangée.
5. Nous répétons de nouveau quatre fois la boucle interne, après la première itération, le résultat reste inchangé, après la seconde, il augmente jusqu'à {A, B, C, E, F}, à la troisième et quatrième, il reste inchangé.
6. Nous répétons de nouveau quatre fois la boucle interne mais *fermeture[C, F]* reste inchangée. L'algorithme est terminé

Nous remarquons que {A, B} n'est pas une super-clé et donc pas une clé candidate.

Corollaire : *Etant donné un ensemble F de dépendances fonctionnelles, on peut facilement déterminer si une dépendance fonctionnelle X $\rightarrow$ Y est dérivable de F, car cette dépendance fonctionnelle sera dérivable si et seulement si Y est un sous-ensemble de la fermeture de X+ de X pour F.*

Par conséquent, nous avons un moyen simple de déterminer si une dépendance fonctionnelle X $\rightarrow$ Y est dans la fermeture F+ de F.

3.2.4. Ensemble irréductible de dépendances

Un ensemble de dépendances fonctionnelles est irréductible si et seulement si il satisfait les trois propriétés suivantes :

1. *Le membre droit (le dépendant) de chaque dépendance fonctionnelle de F contient un seul attribut (Un ensemble singleton).*
2. *Le membre gauche (le déterminant) de chaque dépendance fonctionnelle de F est à son tour irréductible, c'est à dire, aucun attribut ne peut être enlevé du déterminant sans changer la fermeture F+ (sans transformer F en un ensemble qui n'est pas équivalent à F). Nous dirons qu'une telle dépendance fonctionnelle est* ***irréductible à gauche****.*
3. *Aucune dépendance fonctionnelle ne peut être supprimée de F sans changer la fermeture de F+ (sans transformer F en un ensemble qui n'est pas équivalent à F)*

Exemple :

NP $\rightarrow$ NOM
NP $\rightarrow$ MATERIAU
NP $\rightarrow$ POIDS
NP $\rightarrow$ VILLE

Il est facile de voir que cet ensemble de dépendances fonctionnelles est irréductible. Le membre droit est, dans chaque cas, constitué d'un seul attribut. Le membre gauche est à son

tour de façon évidente, irréductible et aucune des dépendances fonctionnelles ne peut être supprimée sans changer la fermeture.

Les ensembles suivants ne sont pas irréductibles :

1. NP $\rightarrow$ {NOM, MATERIAU} : le membre droit n'est pas un singleton
   NP $\rightarrow$ POIDS
   NP $\rightarrow$ VILLE

2. {NP, NOM} $\rightarrow$ MATERIAU : cette dépendance fonctionnelle peut être simplifiée en enlevant NOM du membre gauche sans changer la fermeture.
   NP $\rightarrow$ NOM
   NP $\rightarrow$ POIDS
   NP $\rightarrow$ VILLE

3. NP $\rightarrow$ NP : cette dépendance fonctionnelle peut être supprimée sans changer la fermeture.
   NP $\rightarrow$ NOM
   NP $\rightarrow$ MATERIAU
   NP $\rightarrow$ POIDS
   NP $\rightarrow$ VILLE

Pour chaque ensemble de dépendances fonctionnelles, il existe au moins un ensemble équivalent qui est irréductible.

En effet : Soit *F*, l'ensemble des dépendances fonctionnelles. Grâce à la règle de décomposition, on peut supposer, sans perte de généralité, que chaque dépendance fonctionnelle de *F* a un ensemble droit se réduisant à un singleton. Ensuite, pour chaque dépendance fonctionnelle f de *F* , on examine chaque attribut A du membre gauche de f; si *F* et l'ensemble des dépendances fonctionnelles obtenues en éliminant A du membre gauche de f sont équivalents, on supprime A du membre gauche de f. Alors, pour chaque dépendance fonctionnelle f restante dans *F*, si *F* et *F*- f sont équivalents, alors on supprime f de *F*. L'ensemble *F* final est irréductible et est équivalent à l'ensemble *F* initial.

Exemple :

Soit R (A, B, C, D) et soient les dépendances fonctionnelles
$A \rightarrow BC$, $B \rightarrow C$, $A \rightarrow B$, $AB \rightarrow C$, $AC \rightarrow D$

Nous calculons un ensemble de dépendances fonctionnelles irréductible équivalent :

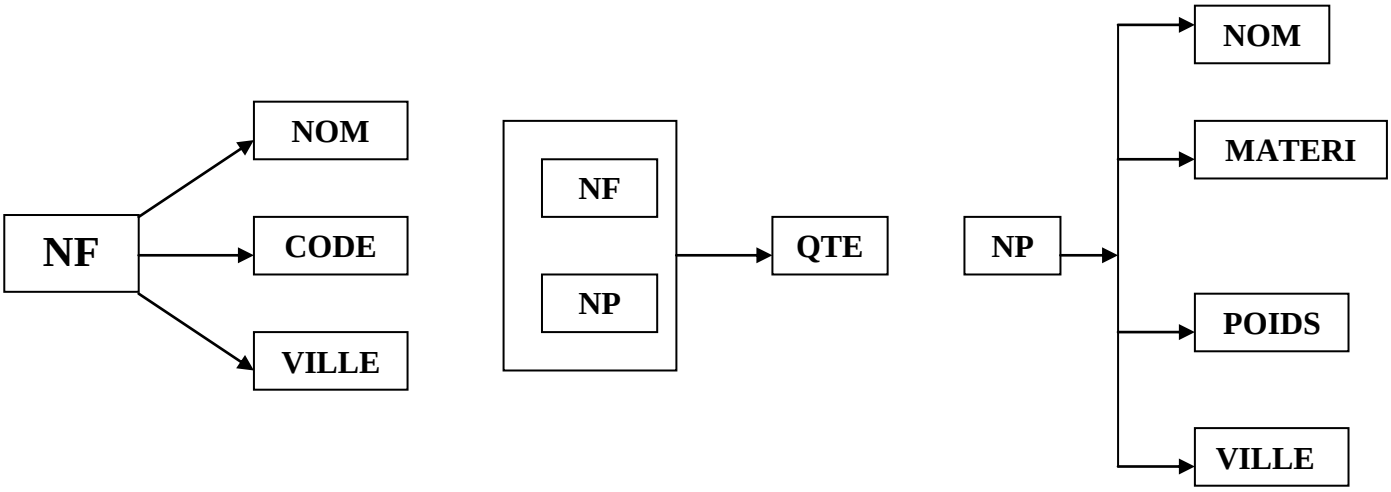
$A \rightarrow B^*$
$A \rightarrow C$
$B \rightarrow C$
$A \rightarrow B^*$
$AB \rightarrow C$
$AC \rightarrow D$

Une des deux occurrences de $A \rightarrow B$ doit disparaître

C peut être éliminée de AC $\rightarrow$ D car nous avons A $\rightarrow$ C et donc A $\rightarrow$ AC (par augmentation) et nous avons AC $\rightarrow$ D et donc A $\rightarrow$ D (par transitivité); ainsi, l'attribut C dans le membre gauche de AC $\rightarrow$ D est redondant.

Nous observons que AB $\rightarrow$ C peut être éliminée, car de nouveau nous avons
A $\rightarrow$ C, donc AB $\rightarrow$ CB (par augmentation), et donc AB $\rightarrow$ C (par décomposition).
A $\rightarrow$ C est impliquée par les dépendances fonctionnelles A $\rightarrow$ B et B $\rightarrow$ C, elle peut également être éliminée. Il nous reste A $\rightarrow$ B, B $\rightarrow$ C, A $\rightarrow$ D.
Cet ensemble est irréductible.

Diagramme de dépendances fonctionnelles



Chaque flèche dans le diagramme est une flèche partant d'une clé candidate (ici clé primaire) de la relation considérée. Par définition, il y aura toujours des flèches partant de chaque clé candidate.

3.3 Normalisation

La conception logique de la base de données utilisée jusqu'à présent semble correcte.

FOURNISSEUR (NF, NOMF, CODE, VILLE) , clé primaire (NF)
PIECE (NP, NOM, CODE, VILLE) , clé primaire (NP)
FOURNITURE (NF, NP, QTE) , clé primaire (NF, NP)

Nous avons vu, qu'intuitivement, le déplacement de VILLE de FOURNISSEUR vers la relation FOURNITURE est une mauvaise conception. La ville du fournisseur concerne le fournisseur et non la livraison.

En fait, les relations dans une base de données relationnelle sont toujours normalisées dans le sens où elles contiennent que des valeurs scalaires. Cependant, une relation donnée pourrait être normalisée dans ce sens et pourtant posséder encore des propriétés indésirables ou des anomalies. Ces anomalies sont :

1. répétition d'informations,
2. inaptitude à la représentation de certaines informations,
3. perte d'information.

La relation VFOURNISSEUR en est un exemple. En effet, nous avions vu que l'adresse d'un fournisseur était répétée pour chacune des pièces fournies, qu'on ne pouvait pas connaître l'adresse d'un fournisseur s'il ne fournit pas de pièce (cas de F5) et enfin qu'on perdait les informations sur un fournisseur s'il ne fournit plus de pièces.

On dit qu'une relation est dans une forme particulière si elle satisfait un ensemble de conditions prédéfinies.

Par exemple, on dit qu'une relation est en première forme normale (abrégé en 1FN), si et seulement si elle satisfait la condition de ne contenir que des valeurs scalaires.

Pour les autres formes, une procédure de normalisation a été proposée, qui consiste à réduire successivement un ensemble de données en une forme plus satisfaisante. La procédure est réversible. Ceci est important car cela indique qu'aucune information n'est perdue.

3.3.1. Décomposition sans perte d'information

Avant d'aborder les problèmes spécifiques de la procédure de normalisation, nous devons examiner le concept de décomposition sans perte d'information. Cette notion est intuitivement liée au concept de dépendance fonctionnelle, et se base sur l'utilisation de l'opérateur de projection. Par ailleurs, pour vérifier qu'une décomposition est sans perte d'information, nous utilisons l'opérateur de jointure entre deux relations. Ces deux opérateurs sont présentés en détail dans le chapitre suivant, mais pour le besoin de ce cours nous les introduisons brièvement dans ce qui suit :

- **Projection**

La projection d'une relation R de schéma R(A1, A2, … , An) sur les attributs Ai_1, Ai_2, … Ai_p avec $i_j \neq i_k$ et $p<n$ est une relation R' de schéma R'(Ai_1, Ai_2, … Ai_p) dont les n-uplets sont obtenus par élimination des valeurs des attributs de R n'appartenant pas à R' et par suppression des n-uplets en double. Cette opération est notée :

$\pi_{Ai1, Ai2, \ldots Aip}$ **(R)**

L'opérateur de projection prend une seule relation comme argument, ainsi qu'un paramètre, qui est la liste des attributs choisi parmi ceux du schéma de la relation donnée en argument.
C'est donc un opérateur pour la construction "verticale" d'un sous-ensemble de la relation R

Exemple :

π_{Ville} **(FOURNISSEUR)**

VILLE
Alger
Tunis
Paris

C'est une relation qui donne les villes où sont localisés les fournisseurs.

- **Jointure naturelle**

Etant donné deux relations R1(A1, … An, X) et R2(X, B1, … , Bm) ayant un ensemble X d'attributs communs (définis sur le même domaine), la jointure naturelle de ces deux relations R1 et R2 notée ∞ est une relation R(A1, … An, X, B1, …, Bn) tel que tout n-uplet t résultant de la composition d'un n-uplet de R1 et d'un n-uplet de R2 ayant les mêmes valeurs pour les attributs X appartient à R.

En d'autres termes si t1 est un n-uplet de R1 et t2 est un n-uplet de R2 et t1[X] = t2[X] alors

$R = R1 \infty R2 = \{ t / t = t1 \cup X \cup t2 : (t1 \cup X \in R1) , (X \cup t2 \in R2) \}$

Exemple :

PIECE ∞ $\pi_{NF, NP}$(FOURNITURE)

N P	NOM	MATERIAU	POIDS	VILLE	NF
P 1	Vis	Fer	12	Alger	F1
P1	Vis	Fer	12	Alger	F2
P2	Boulon	Acier	17	Tunis	F1
P 2	Boulon	Acier	17	Tunis	F2
P2	Boulon	Acier	17	Tunis	F3
P 3	Ecrou	Zinc	17	Paris	F1
P4	Ecrou	Fer	14	Alger	F1
P 4	Ecrou	Fer	14	Alger	F4
P5	Came	Zinc	12	Tunis	F1
P 5	Came	Zinc	12	Tunis	F4
P6	Clou	Fer	19	Alger	F1

C'est une relation qui donne les informations sur les pièces ainsi que le numéro des fournisseurs qui les fournissent. C'est le résultat d'une jointure entre PIECE et FOURNITURE sur l'attribut commun NF.

Considérons la relation fournisseur réduite uniquement aux attributs (NF, CODE, VILLE) avec l'extension suivante :

NF	CODE	VILLE
F3	130	Tunis
F5	130	Oran

Soit les deux décompositions possibles de cette relation

a)

FC : $\pi_{NF, CODE}$(Fournisseur)

NF	CODE
F 3	130
F5	130

FV : $\pi_{NF,VILLE}$(Fournisseur)

NF	VILLE
F3	Tunis
F5	Oran

b)

FC : $\pi_{NF, CODE}$(Fournisseur)

NF	CODE
F3	130
F5	130

CV : $\pi_{CODE, VILLE}$(Fournisseur)

CODE	VILLE
130	Tunis
130	Oran

Nous constatons que

1. dans le cas a) aucune information n'est perdue ; les deux relations FC et FV nous indiquent que le fournisseur F3 a le code 130 et Tunis comme adresse et le fournisseur F5 a un code 130 et Oran comme adresse. Cette décomposition est bien sans perte d'information.
2. Dans le cas b), au contraire, des informations sont définitivement perdues ; nous pouvons encore dire que les deux fournisseurs ont un code de 130, mais nous ne pouvons plus dire quelle est la ville de chaque fournisseur. La seconde décomposition est une décomposition avec perte d'information.

D'où vient cette différence ? Nous remarquons que la décomposition est une projection de la relation Fournisseur. Nous remarquons aussi que dans le cas a) aucune information n'est perdue car si nous effectuons la jointure des relations FC et FV, nous obtenons de nouveau la relation initiale fournisseur. Dans le cas b) au contraire, si nous effectuons la jointure des deux relations FC et CV, nous n'obtenons pas de nouveau la relation initiale fournisseur, et par conséquent nous avons perdu des informations.

La réversibilité signifie que la relation initiale est égale à la jointure de ses projections.

D'une manière générale, quelles sont les conditions à satisfaire pour que la jointure des deux décompositions nous permet de retrouver la relation initiale ?
Dans notre exemple, la relation fournisseurs satisfait l'ensemble irréductible de DFs

NF $\rightarrow$ CODE
NF $\rightarrow$ VILLE

Et cette relation est égale à la jointure de ses projections sur (NF, CODE) et (NF, VILLE)

Principe de décomposition :

Théorème de Heath : *Soit la relation R(A, B, C) où A, B et C sont des ensembles d'attributs. Si R satisfait la DF A*$\rightarrow$B, alors R est égale à la jointure de sa projection sur {A, B} et {A, C}

Dans notre exemple, nous constatons que pour la seconde décomposition (décomposition b) la DF : NF $\rightarrow$ VILLE est perdue.

Qualité d'une décomposition :

La qualité d'une décomposition est liée à deux aspects :

1. être sans perte d'information, i.e., la jointure doit être conservative,
2. être préservative des DFs

Préservation des DFs :

Soit **R** une relation et **F** son ensemble de DFs, et soit une décomposition R1, R2, …, Rn de R ; cette décomposition préserve les DFs si et seulement si la fermeture F^+ de F est identique à celle de l'union des fermetures des {Ri, i =1,…,n}.
C'est-à-dire **G= {F1} U {Fi} U … U {Fn} et $F^+ = G^+$**

3.3.2. Première, seconde et troisième formes normales

Définition 7. Relation 1FN. Nous rappelons qu'*une relation est en 1 FN si tous les domaines sous-jacents contiennent uniquement des valeurs scalaires.*

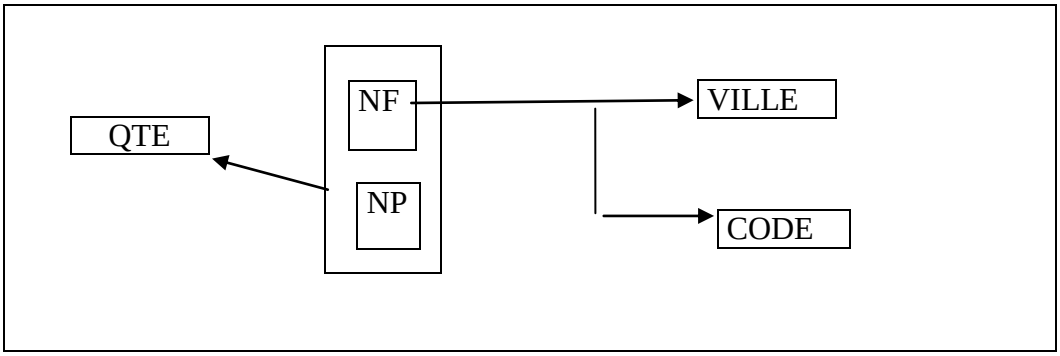
Une relation qui est seulement 1FN a une structure qui n'est pas satisfaisante pour un certain nombre de raisons.

Exemple :
Supposons que les deux relations Fournisseur et Fourniture soient réunies en une seule relation appelée Première.

PREMIERE (NF, CODE, VILLE, NP, QTE)
Clé primaire (NF, NP)

Pour les besoins de l'exemple, nous introduisons une contrainte supplémentaire
VILLE → CODE

La signification de cette contrainte est que le code d'un fournisseur est déterminée par la ville où il habite. Par exemple, les fournisseurs d'Alger ont un code 120. Pour simplifier l'exemple, nous avons omis l'attribut NOM du fournisseur.
Le diagramme des DFs de la relation Première est :



Remarques : les flèches dans une relation qui n'est qu'en 1FN sortent de partout. Ceci est un signe de problème. Pour illustrer ces problèmes, considérons l'extension suivante de la relation Première :

PREMIERE

NF	CODE	VILLE	NP	QTE
F1	120	Alger	P1	300
F1	120	Alger	P2	200
F1	120	Alger	P3	400
F1	120	Alger	P4	200
F1	120	Alger	P5	100
F1	120	Alger	P6	100
F2	110	Tunis	P1	300
F2	110	Tunis	P2	400
F3	110	Tunis	P2	200
F4	120	Alger	P2	200
F4	120	Alger	P4	300
F4	120	Alger	P5	400

La seule modification au niveau des données est le code de F3 qui est 110 au lieu de 130 dans l'exemple initial, pour être cohérent avec la DF.
Les redondances sont évidentes. La ville d'Alger est répétée pour chaque fournisseur F1 de même que le code. Ces redondances, nous l'avons déjà vu, génèrent des problèmes appelés les **anomalies de mise à jour**.

1. Nous ne pouvons pas insérer un fournisseur qui habite une ville tant que le fournisseur ne fournit pas au moins une pièce. Cas du fournisseur F5 qui habite Oran.
2. Si nous supprimons le n-uplet ayant F3 comme valeur d'attribut, nous perdons l'information que F3 habite Tunis.

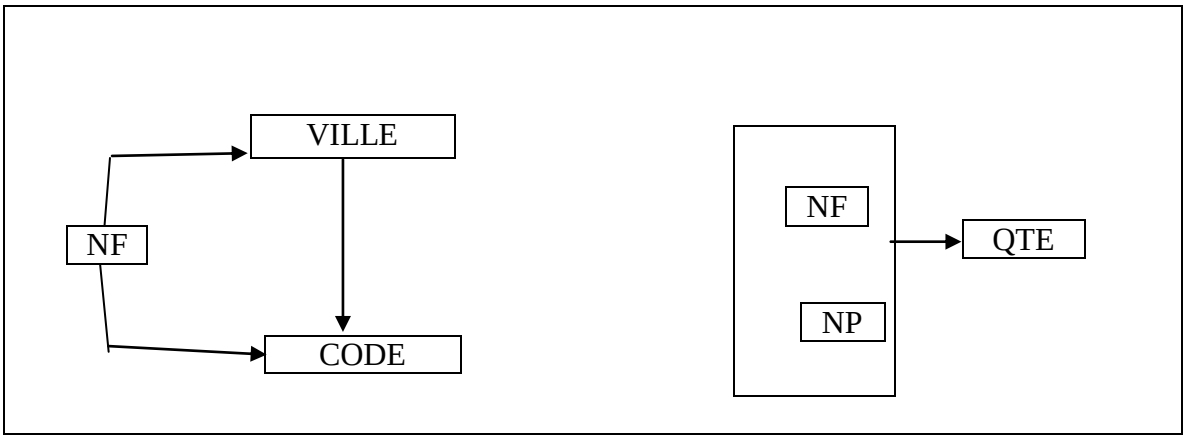
3. Si le fournisseur F1 déménage d'Alger vers Annaba, la procédure est longue et risque de poser des problèmes si elle n'est pas automatisée.

Remarques : Ces problèmes sont dus au fait que la relation Première contient trop d'informations regroupées. Des informations sur des fournisseurs et sur leurs livraisons. Lorsqu'on supprime des informations sur une livraison, on perd également les informations sur les fournisseurs.

La solution à ces problèmes est de remplacer la relation Première par deux relations

SECONDE (NF, CODE, VILLE)
FOURNITURE (NF, NP, QTE)

Avec le diagramme des DFs



Et l'extension est :

SECONDE

NF	CODE	VILLE
F1	120	Alger
F2	110	Tunis
F3	110	Tunis
F4	120	Alger
F5	130	Oran

FOURNITURE

NF	NP	QTE
F1	P1	300
F1	P2	200
F1	P3	400
F1.	P4	200
F1	P5	100
F1	P6	100
F2	P1	300
F2	P2	400
F3	P2	200
F4	P2	200
F4	P4	300
F4	P5	400

Les informations concernant les fournisseurs ont été regroupées dans Seconde, par conséquent, les informations concernant F5 sont dans seconde et pas dans fourniture. Avec ces nouvelles relations, les problèmes précédents sont résolus.

Définition 8. Relation 2FN. *Une relation est en 2FN si et seulement si elle est en 1FN et si chaque attribut non clé est en dépendance irréductible avec la clé.*

Les relations Seconde et fourniture sont en 2FN. Les clés sont NF pour Seconde et (NF, NP) pour Fourniture. La relation Première n'est pas en 2FN.

Une relation qui est 1FN et pas 2FN peut être décomposée en un ensemble de relations 2FN équivalentes (sans pertes d'informations).

Résumé :

Une première étape de la procédure de normalisation consiste à effectuer des projections pour supprimer les dépendances fonctionnelles qui ne sont pas irréductibles. Ainsi, soit la relation

R (A, B, C, D) avec la clé primaire (A, B) et la DF A $\rightarrow$ D

Les principes de normalisation recommandent de remplacer R par ses deux projections R1 et R2 avec :

R1 (A, D) et la clé primaire A
R2 (A, B, C) et la clé primaire (A, B)

La relation R peut être retrouvée en effectuant la jointure de R1 et R2 sur la clé de R2.
Cette décomposition pose néanmoins des problèmes dans le cas de la relation Seconde. En effet, la dépendance CODE sur NF, bien qu'elle soit fonctionnelle et en fait irréductible, est transitive (via VILLE). Chaque valeur de ville à son tour détermine une valeur de CODE.

D'une manière générale, lorsque les DFs A $\rightarrow$ B et B $\rightarrow$ C sont valides alors A $\rightarrow$ C est valide aussi, et les dépendances transitives posent aussi des problèmes lors des mises à jour.

Exemple :

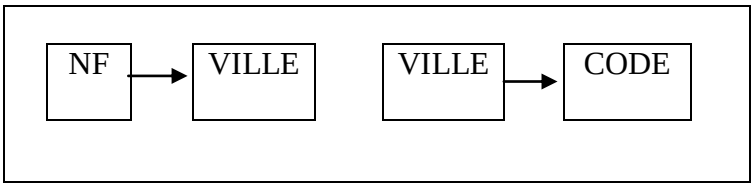
Concernant la redondance VILLE – CODE correspondant à la DF VILLE $\rightarrow$ CODE

- On ne peut pas exprimer le fait qu'une ville particulière a un code particulier tant qu'il n'y a pas de fournisseur qui habite dans cette ville.
- S'il n'y a qu'un seul n-uplet pour une ville particulière de la relation Seconde, et si on le supprime, on supprime non seulement l'information pour le fournisseur concerné mais également le fait que cette ville a un code particulier.
- La valeur du code pour une ville donnée apparaît en général, plusieurs fois dans la relation Seconde, ce qui pose problème lors des mises à jour.

De nouveau, la solution à ces problèmes est de remplacer la relation initiale (Seconde) par deux projections :

FV (NF, VILLE)
VC (VILLE, CODE)

Le diagramme des DFs est :



FV

NF	VILLE
F1	Alger
F2	Tunis
F3	Tunis
F4	Alger
F5	Oran

VC

VILLE	CODE
Alger	130
Oran	120
Tunis	110
Annaba	150

L'opération est réversible puisque Seconde est la jointure de FV et VC sur VILLE. L'effet de cette normalisation a été d'éliminer la dépendance transitive de CODE sur NF.
Le problème ici est que le code ne décrit pas le fournisseur mais la ville dans laquelle habite le fournisseur.

Définition 9. Relation 3FN. *Une relation est en 3FN si et seulement si elle est en 2FN et si chaque attribut non clé est en dépendance non transitive avec la clé primaire.*

Les relations FV et VC sont en 3FN.

Résumé : Effectuer des projections pour éliminer les dépendances transitives.
Si R(A, B, C) et A est une clé primaire et B $\rightarrow$ C est une DF .
Les principes de normalisation recommandent de remplacer R par ses deux projections R1 et R2 de la façon suivante
R1 (B, C) avec B comme clé primaire
R2 (A, B) avec A comme clé primaire
La relation R peut être retrouvée en effectuant la jointure de R1 et R2 sur la clé étrangère de R2 qui est aussi la clé primaire de R1.

Il a été prouvé que toute relation a une décomposition en 3FN sans perte d'informations et qui préserve les DFs. Plusieurs approches permettent cette décomposition : nous présenterons une approche basée sur l'algorithme de Berstein.

Algorithme de synthèse (BERNSTEIN)

Cet algorithme est basé sur la décomposition de la relation universelle. **La relation universelle** est une relation définie sur tous les attributs.

ENTREE : R = {A, F}, A = {ensemble de tous les attributs}, F = {ensemble des DFs}
SORTIE : D = {R1, R2,…, Rn} tel que Ri = {Xi, Y_i }$_{i=1,…,n}$ en 3FN
ETAPE 1 : Rechercher une couverture irréductible de F notée F'
ETAPE 2 : Partitionner F' en groupes F'$_1$, F'$_2$,…, F'$_k$ tels que toutes les DFs d'un même groupe aient même partie gauche Xi
ETAPE 3 : Pour chaque groupe F'i construire un schéma Ri={Xi, Yi} où Yi est l'ensemble de tous les attributs apparaissant à droite du groupe F'i
ETAPE 4 : Soient A1, A2, An les attributs de R (s'il y en a qui n'ont pas encore été pris en compte. Ajouter à D le schéma de relation constitué de la projection de R sur ces attributs.
ETAPES 5 : Si aucune relation de D n'inclut une clé candidate de R, ajouter à D le schéma de relation constitué d'une clé candidate de R.

3.3.3 Forme normale de Boyce/ Codd

Dans cette partie, il est remis en cause l'hypothèse simplificatrice que chaque relation a une seule clé candidate. En fait le cas général impose qu'une relation peut avoir plusieurs clés candidates. Nous traiterons dans ce paragraphe le cas général où :

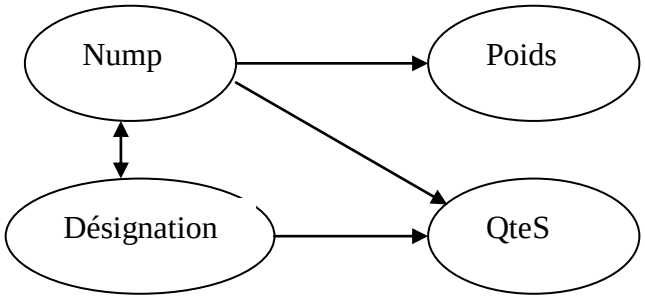
- Une relation peut avoir plusieurs clés candidates,
- Les clés candidates peuvent être composées
- Elles se chevauchent, c'est-à-dire qu'elles ont au moins un attribut commun.

Nous substituons alors à la troisième forme normale une forme plus forte appelée **forme normale de Boyce/ Codd ou BCNF :**

Définition 10. Relation en BCNF. *Une relation est en BCNF si et seulement si les seuls déterminants sont des clés candidates.*

La BCNF impose qu'il ne peut pas exister dans une relation, d'autres dfs que celles qui partent des clés candidates.

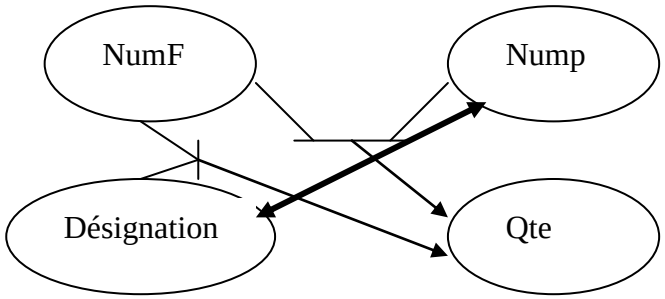
Exemple1 :
Produit (NumP, Designation, Poids, QteS) et NumP et Désignation sont deux clés candidates



Cette relation est en BCNF, car les seules flèches partent des deux clés candidates.

Exemple2 :
Four-produit (Numf, Nump, Designation, Qte)

Les clés candidates sont (NumF, Nump) et (Numf, Désignation) : Cette relation est-elle en BCNF ?



Cette relation n'est pas en BCNF, car il existe deux flèches entre des parties de clés.

Soit une extension de cette relation :

Numf	***NumP***	***Désignation***	***Qte***
F1	***P1***	***clou***	***100***
F2	***P1***	***clou***	***200***
F3	***P1***	***clou***	***400***
F4	***P1***	***clou***	***100***

La redondance désignation avec Nump, introduit une anomalie de mise-à-jour : en effet si on veut changer la désignation pour le produit P1, il faut rechercher toutes les occurrences dans la table sinon il y aurait incohérence.
Solution : décomposition en deux relations en BCNF:
R1(Nump, désignation) et R2 (Numf, Nump, Qte)
Ou bien :
R1 (Nump, désignation) et R2 (Numf, Désignation, Qte)