COURS DE BASES DE DONNÉES

Année 1999-2000

S.Grigorieff

Cours de Base de données

07.02.2000

Introduction

- La base de données est une collection d'information (pouvant être très grosse) + un mode d'organisation et de gestion

- Le système gérant l'ensemble est appelé système de gestion de la base de données SGBD (DBMS Date Base Management Systeme)

- Un SGBD est un ensemble coordonné de logiciels qui permet :
  1. Spécifier un modèle de BD et de le gérer.
  2. Créer une BD (en déchargeant l'utilisateur des problèmes d'implantation physiques des données).
  3. Interroger la BD (on parle de requête, query) et manipuler les données en optimisant les coûts.
  4. Assurer la cohérence de la base (on dit aussi intégrité) alors que plusieurs utilisateurs peuvent y accéder simultanément.
  5. Assurer sécurité et confidentialité.

Dans ce cours on étudiera le modèle relationnel, et l'on utilisera le langage SQL (qui est à la fois un langage de requête et capable de gérer le modèle lui-même).

« Modèle relationnel »: c'est un modèle de l'implantation des données.

Avant de passer à un modèle relationnel on cherche à comprendre et visualiser l'organisation fonctionnelle des données.
Il faut spécifier un modèle sémantique :
Nous utiliserons le modèle Entité/relation (ou Entité/Association) modèle proposé par I.Chen en 1977

Chapitre 1. Le modèle Entité/Relation (ou Entité/Association)

1) Les objets du modèle Entité/Relation

1. les entités (objets de base)

Pour I.Chen ce sont des objets que l'on peut identifier distinctement.
Par exemple : l'élève Dupond, le professeur Durand, l'amphi X1,le cours de BD.

2. Les types d'entités (ou ensemble d'entités)

Ce sont des groupes d'entités ayant quelque similarité

3. Les associations

Donnons un exemple :
« l'étudiant E étudie la matière M », cette phrase exprime le type d'association « étudie » entre les types d'entités « étudiant » et « matière ».

4. Les attributs

Un attribut est une propriété qui associe à chaque entité d'un type d'entités, ou d'une association de types d'entités, une valeur dans un certain domaine (nombres / chaînes de caractères).

Exemples :

Le type d'entités « étudiant » a pour attributs → Nom, Numéro d'étudiant, Adresse, Diplôme préparé

L'association « le fournisseur F fournit l'article A » peut avoir comme attribut le prix.

5. Les clés

Une clé est un attribut ou un groupe d'attributs dont la valeur doit identifier de façon unique une entité. On distingue une clé principale (ex. le numéro de sécurité sociale qui permet d'identifier totalement un individu)

Attention : Déclarer un attribut comme clé est une **décision** du concepteur de la base de données et non pas le constat fait sur un état à un moment de la BD.
Cette décision est une contrainte.

6. Les sous-types

Un sous-type hérite automatiquement des attributs du sur-type et il peut avoir des attributs spécifiques qui n'ont pas de sens pour les autres entités du sur-type.

Par exemple : garçon est un sous type d'individu et peut avoir comme attribut « état vis à vis du service militaire » dont les valeurs sont : dégagé, sursitaire, exempté, réformé.

7. La fonctionnalité des associations

L'association entre E_1, E_2 ,……, E_n est fonctionnelle de E_1, E_2 , …, E_{i-1}, E_{i+1},……, E_n vers E_i si :
Quelque soient $x_1 \in E_1$,……, $x_n \in E_n$ $(x_1,……,x_n)$ associés
$y_1 \in E_1$,……, $y_n \in E_n$ $(y_1,……,y_n)$ associés
Si $x_1=y_1$et …… et $x_{i-1}= y_{i-1}$ et $x_{i+1}= y_{i+1}$ et …… et $x_n= y_n$ alors $x_i = y_i$

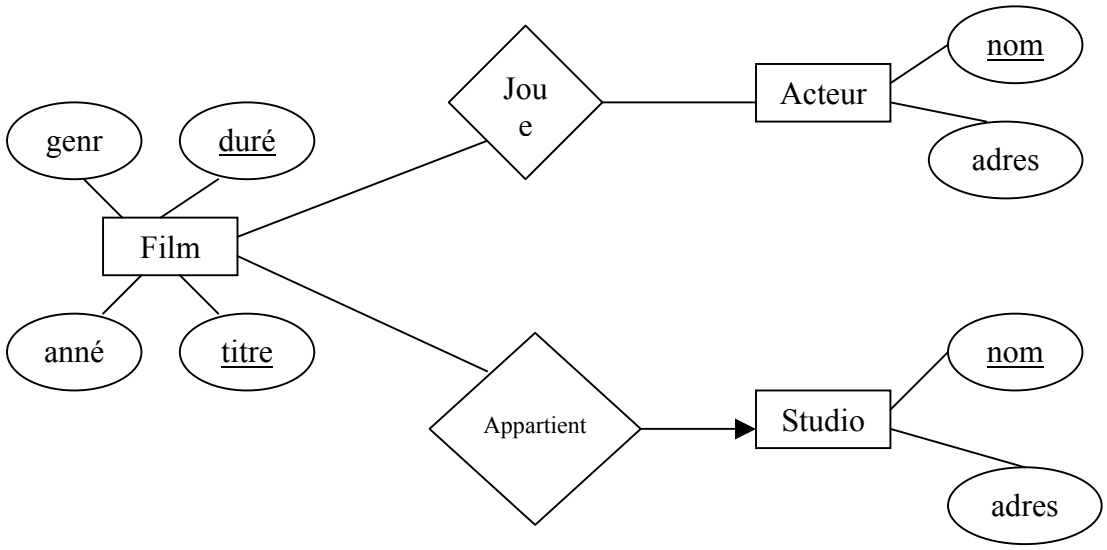
Attention : Déclarer un attribut comme clé est une **décision** du concepteur de la base de données.

2) Diagramme Entité/Relation

Conventions :

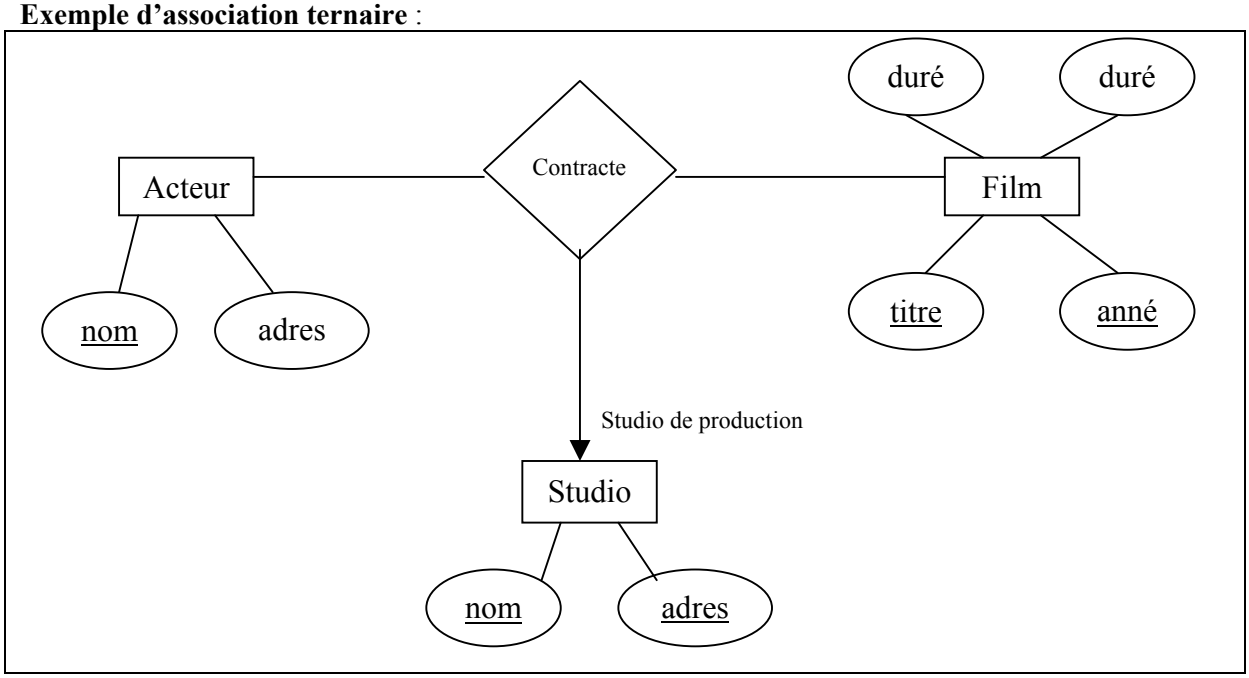
- Types d'entités → rectangle
- Attribut → ovale
- Association → losange + traits le reliant aux types en jeu
- On souligne les attributs constituant la clé principale.
- Lorsqu'il y a fonctionnalité d'une association on met une flèche orienté vers Ei

Exemple :

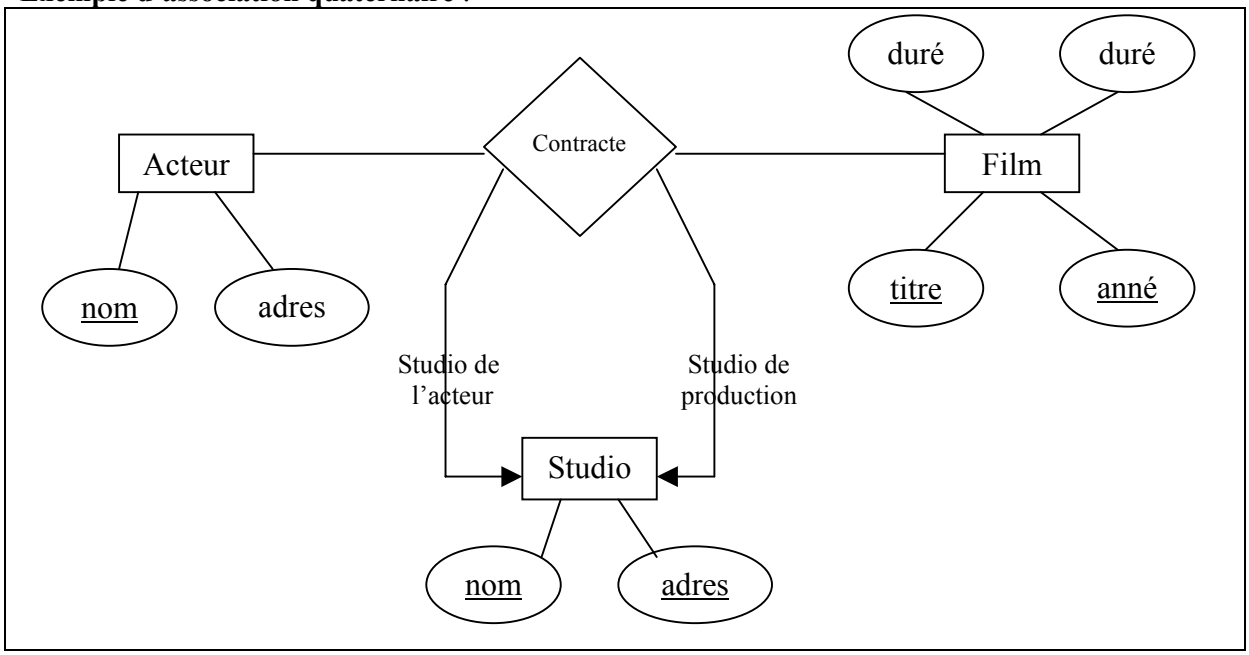


On a l'association fonctionnelle Film, Studio (un film est produit par un seul studio).

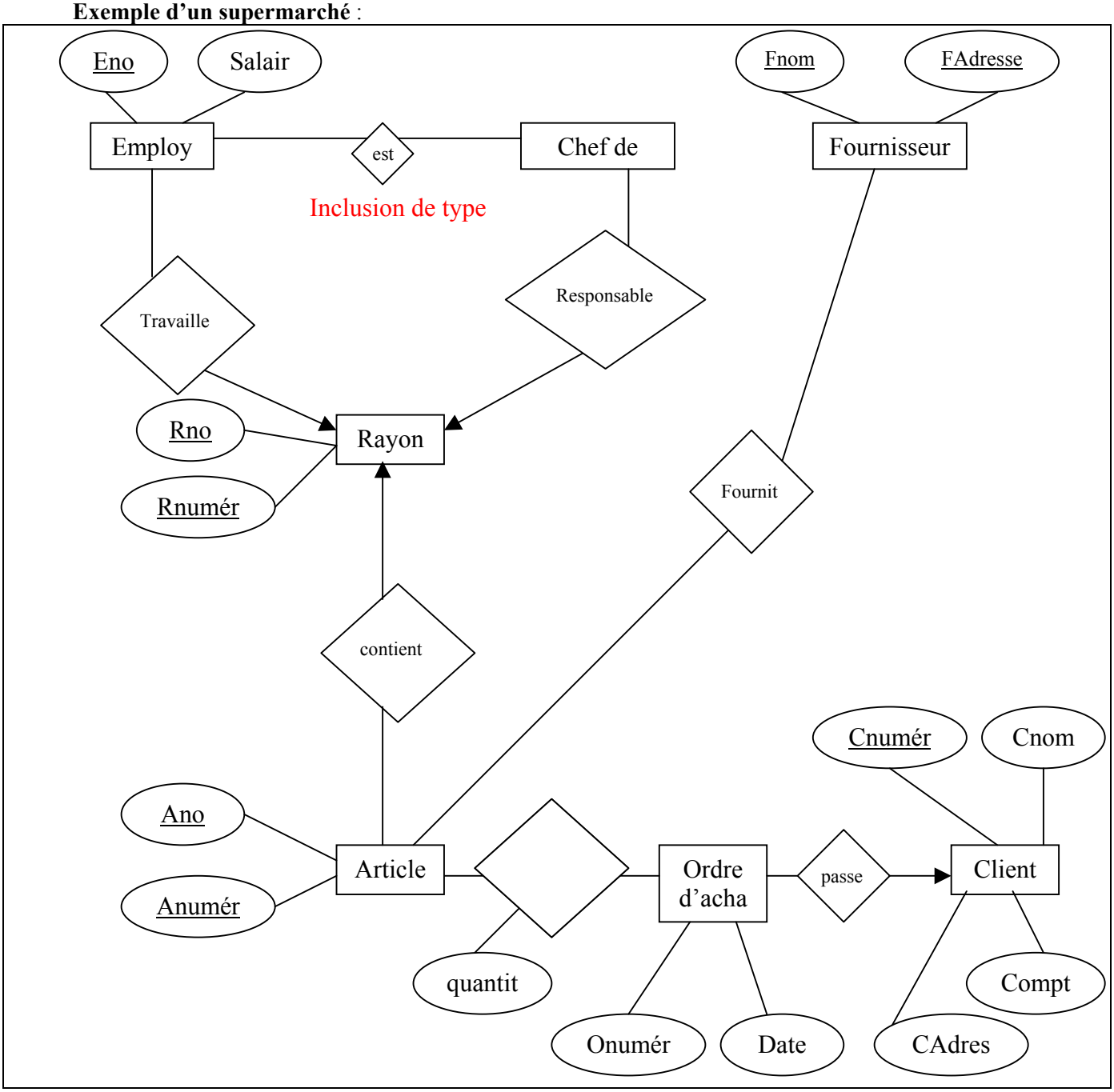
Exemple d'association ternaire :



Exemple d'association quaternaire :



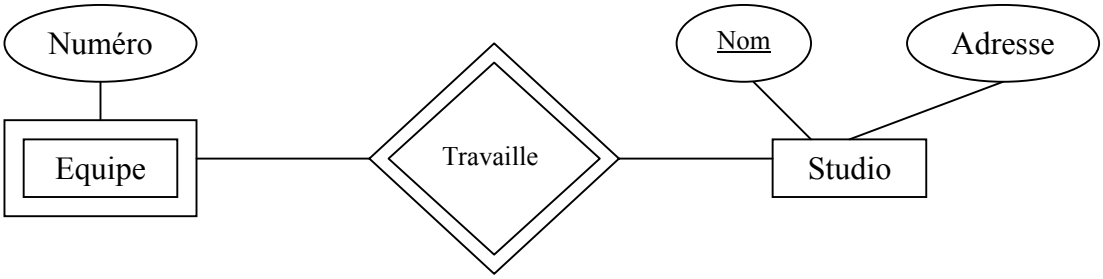
Exemple d'un supermarché :



3) Type faible ou dépendant d'entités

Ce sont des types qui n'ont pas de clé propre mais une clé obtenue comme suit :
- des attributs du type faible E
- les attributs constituant les clés de types qui sont chacun associés de façon fonctionnelle à E

Soit l'entité faible E
Soient :
Les entités F_1, ……, F_n
Les relations :
R_1 fonctionnelle de E vers F_1
R_2 fonctionnelle de E vers F_2
⋮ ⋮
R_n fonctionnelle de E vers F_n
Les clé de F_1,……F_n



Une clé pour l'entité faible « Equipe » est Numéro, Nom

Convention :
On double les traits du rectangle de l'entité faible et des losanges des associations fournissant la clé.

Cours de Base de données

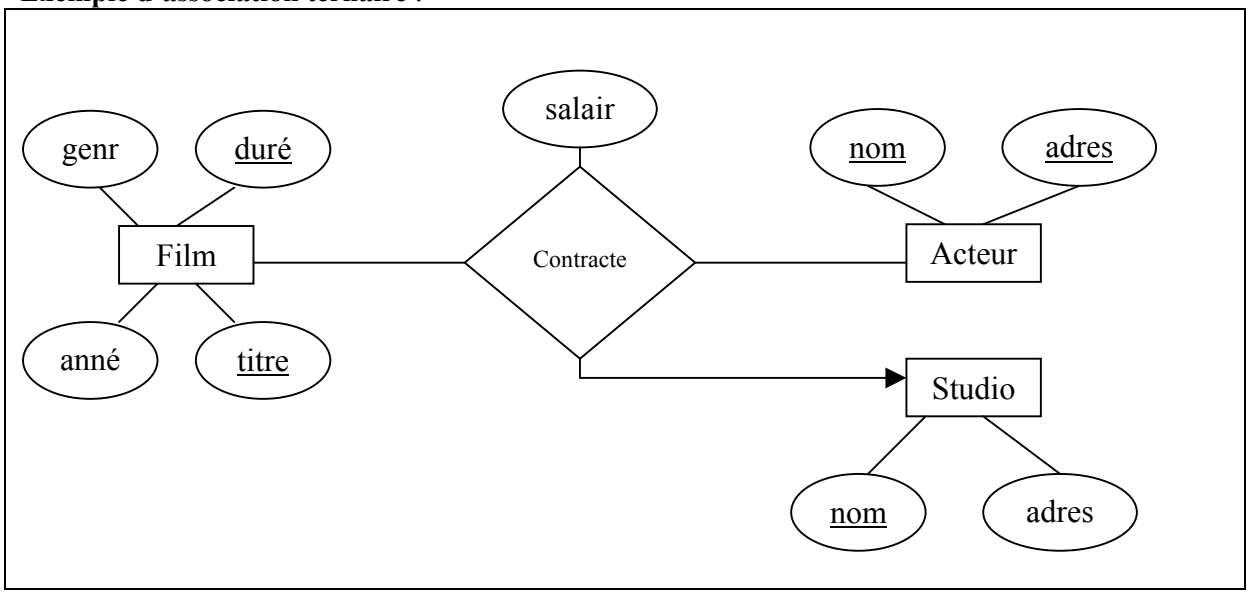
14.02.2000

- Rappel du cours précédent :

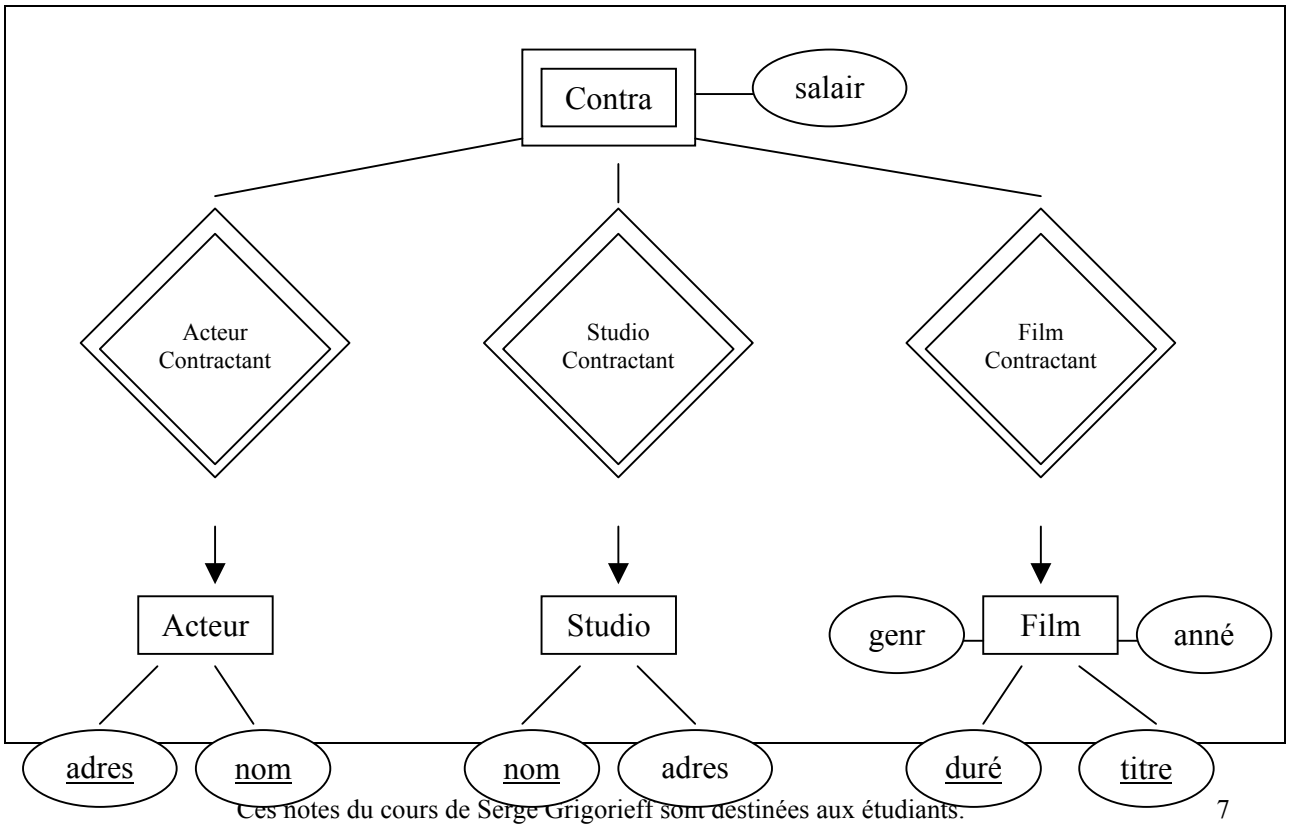
Type T d'entités faible : il n'a pas de clé propre, il faut prendre des clés d'autres types T_1, ……, T_n pour lesquels il existe des associations fonctionnelles de T vers les T_i.

De telle entités faible s'introduisent quand on veut éclater une association d'arité (nombre de composants) ≥3 en une famille d'associations binaires fonctionnelles.

Exemple d'association ternaire :



Cette association se traduit en :



Chapitre 2. Le modèle relationnel (E.F. Codd 1970 IBM)

Buts de ce modèle:
- Indépendance des programmes d'applications et des sessions interactives vis à vis de la représentation interne des données.
- Base théorique solide pour traiter des problèmes de cohérence et de redondance.

1) Les tables ou le modèle relationnel tel qu'il apparaît à l'utilisateur

Pour un SGBD relationnel on a les propriétés suivantes :

1. Il n'y a que des tables

- Un table consiste en un schéma relationnel qui comporte :
  - Le nom de la table,.
  - Un nom (appelé aussi attribut de la table) et un type (scalaire) pour chaque colonne.
- Des contraintes de fonctionnalité, on décrète que tel attribut ou groupe d'attributs G est clé de la table. On déclare les clés candidates et les clés principales.
- Un contenu : des lignes pour lesquels l'élément contenu en colonne A est dit type associé à l'attribut A.

Exemple de table :
Voie_Publique

Nom	Max_impair	Max_pair	Quartier	Arrondissement
Rue Jussieu	45	4	Saint Victor	5
Rue Linné	45	24	Saint Victor	5
Rue Chevaleret	199	148	Gare	13
Rue Montaigne	63	60	Champs Elysées	8
Rue de Tournon	33	20	Odéon	6

Les nom des colonnes sont deux à deux distincts.

2. L'ordre des lignes et celui des colonnes est ignoré

a) Ordre des colonnes

C'est un ordre sur les attributs qui n'est pas pertinent. On ne peut référencer une colonne que par son nom (attribut).

b) Ordre des lignes

Il traduit une évolution temporelle de la table donc cet ordre n'est pas pertinent.

On voit donc qu'il n'est pas possible de référencer une ligne.
Mais 2 lignes distinctes doivent différer sur au moins un attribut.

3. Il y a des opérateurs

Il sont à disposition de l'utilisateur, ils génèrent des tables à partir d'autres tables.

• Création d'une table en SQL (Structured Query Language)

```
CREATE TABLE Voix_Publique
 ( Nom VARCHAR
  Max_impair    INTEGER
  Max_pair      INTEGER
  Quartier      VARCHAR
  Arrondissement INTEGER
  PRIMARY KEY(Nom) )

INSERT INTO Voix_Publique VALUES ('rue jussieu',45,4,'Saint
Victor',5)
```

2) Modélisation mathématique des tables par les relations

C'est à dire les sous ensemble de produit cartésiens ou encore les ensembles de listes toutes de même taille.

Soit T une table avec des attributs A_1, ……, A_n de domaines associés D_1, ……, D_n .
On lui associe une relation $R \subseteq D_1 \times D_2 \times \ldots\ldots D_n$.
R est l'ensemble des n-uplet $(x_1, \ldots\ldots, x_n)$ tels que la table T contienne une ligne ayant :
- en colonne A_1 la valeur x_1.
  ⋮ ⋮
- en colonne A_n la valeur x_n.

Les lignes de T sont des n-uplets de R
Les colonnes de T sont des composantes de R (projection de R sur ces composantes)

Un groupe d'attributs $G=\{Ai_1, \ldots\ldots Ai_k\}$ de T est une clé de T, si et seulement si la relation R est fonctionnelle par rapport aux composantes $i_1, \ldots\ldots, i_k$.

• Formalisme simple sur une théorie mathématique riche.

L'ordre (contingent à la représentation tabulaire) sur les lignes de T a disparu dans cette formalisation car il n'y a pas d'ordre intrinsèque sur les n-uplets de R.

Hélas l'ordre des colonnes lui reste bien présent et intrinsèque, ce qui est fâcheux.

Soit X,Y deux ensembles.

$X \times Y \neq Y \times X$ sauf si $X=Y$ ou $X=\emptyset$ ou $Y=\emptyset$
(a,b) $a,b \in R$ $(a,b) \neq (b,a)$
$R \subseteq X \times X$
$\{(a,b) \in X \times X \quad (a,b) \in R\} \neq \{(b,a) \in X \times X \quad (a,b) \in R\}$

3) Modélisation mathématique par les ensembles de fonctions de même domaine

Soit T une table avec des attributs A_1, ……, A_n de domaines associés D_1, ……, D_n .
On lui associe un ensemble F de fonctions toutes de domaine { A_1, ……, A_n }.
Toutes les fonctions de $f \in F$ vérifient $f(A_1) \in D_1$ …… $f(A_n) \in D_n$.

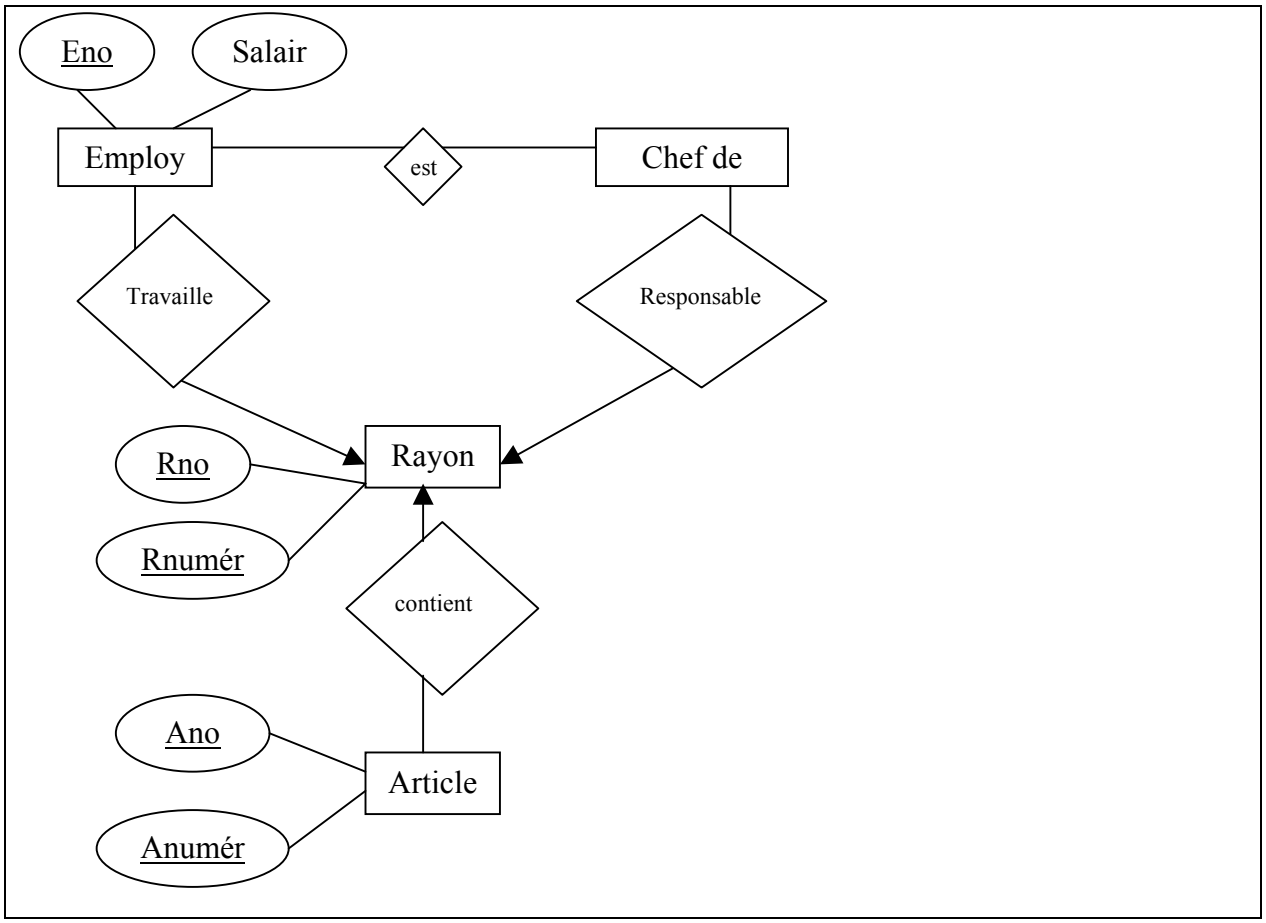
En fait :
$f \in F \Leftrightarrow$ il existe une ligne de la table T dont :
l'élément en colonne A_1 est exactement $f(A_1)$
⋮ ⋮
l'élément en colonne A_n est exactement $f(A_n)$

- Les lignes de T sont exactement les f de F (vues de façon déroulés) on écrit $f(A_i)$ en colonne A_i, l'ordre des lignes de T a donc disparu car il n'y a pas d'ordre intrinsèque sur l'ensemble F.

- La colonne A_i de T correspond aux $f(A_i)$, f variant dans F. Les colonnes T correspondent donc aux éléments du domaine (ensemble sans ordre intrinsèque) commun des fonctions dans F.
On a perdu l'ordre sur les colonnes.

4) traduction d'un diagramme Entité/Relation dans le modèle relationnel

1. Un type d'entités (pas faible) se représente par une table, les attributs de la table sont les attributs du type d'entités, les domaines associés sont les types scalaires associés à ces attributs dans le diagramme E/R.



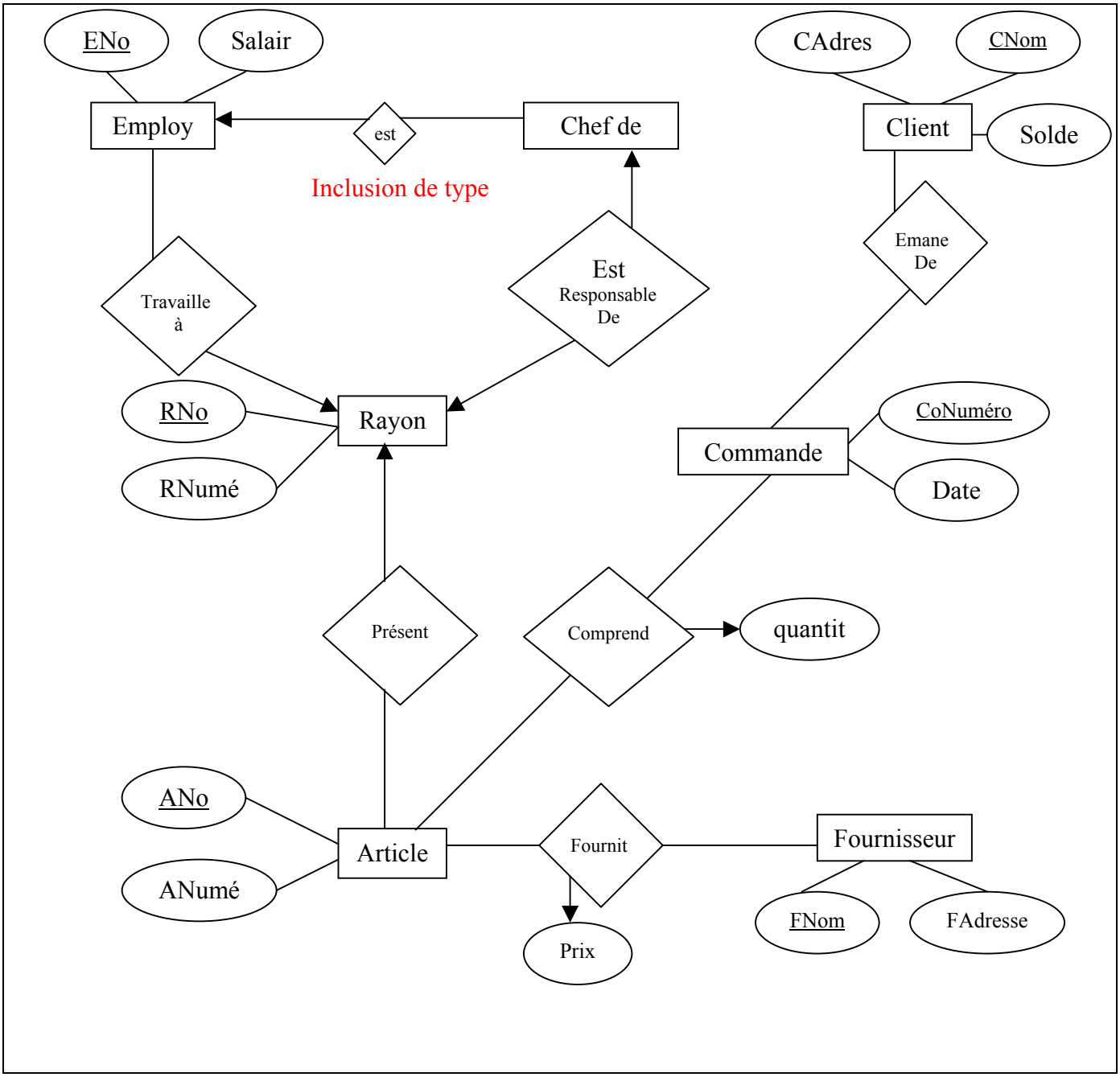
Tables associés :
Employé(ENom, salaire)
Chef de Rayon (ENom)
Rayon (RNom, RNuméro)
Article (ANom, ANuméro)

2. un sous type E d'un type F se traduit par une table dont les attributs sont ceux du type E et ceux de la clé principale du type F. Par exemple :Chef de Rayon

Cours de Base de données

21.02.2000

Reprenons l'exemple du supermarché dont le diagramme Entité/Relation est le suivant :



4) Traduction d'un diagramme Entité/Relation dans le modèle relationnel.

- **Type d'entité (non faible)**

Pour un type d'entité (non faible) on associe une table qui possède les propriétés suivantes :
- Les attributs de la table sont les noms des attributs du type d'entités.
- Le domaine d'un attribut de la table est celui associé à l'attribut correspondant du type d'entités.

Exemple :

1. Employé(ENom, Salaire)
2. Rayon(RNom, RNuméro)
3. Article(ANom, ANuméro) — ANuméro signifie que c'est une clé potentiel
4. Fournisseur(FNom, FAdresse)
5. Commande(CoNuméro, Date)
6. Client(CNom, CAdresse, Solde)

- **Sous type E d'un type F**

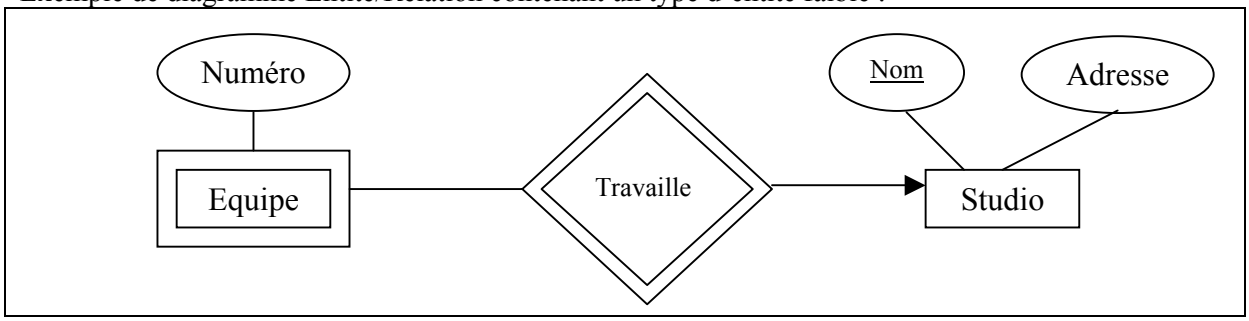
On lui associe une table dont les attributs sont les noms des attributs de E et ceux des attributs de la clé principale de F, les domaines sont ce qu'on pense.

Exemple :

7. Chef de rayon(ENom)

- **Type d'entité faible de E**

Exemple de diagramme Entité/Relation contenant un type d'entité faible :



On associe au type d'entité faible une table d'attributs contenant tous ceux de E, ainsi que ceux des clés principales des types E_1, ……, E_n auquel E est relié de façon relationnelle.

- **Association**

On lui associe une table dont les attributs sont ceux des clés principales de F_1, ……, F_k (après un renommage éventuel), et les attributs de l'association elle-même.

Exemple :

8. Travaille à(ENom, RNom)
9. Est Responsable De(ENom, RNom)
10. Présent(ANom, RNom)
11. Fournit(FNom, ANom, Prix) — Ici il n'y a pas de relation fonctionnelle et la clé a 2 attributs
12. Comprend(CoNuméro, ANom, Quantité) — Ici il n'y a pas de relation fonctionnelle et la clé a 2 attributs
13. Emane De(CoNuméro, CNom)

Pour obtenir une clé de cette table on considère une clé pour chacun d'entre les F_1, ……, F_k et les attributs de l'association elle-même. Si l'association est fonctionnelle vers F_i, la clé de F_i est inutile, de même tous les attributs fonctionnels de l'association sont aussi inutiles.

Relations avec une clé commune.

Il y a une simplification possible d'une BD :
Si deux tables ont une clé commune alors il est préférable de les joindre en une seule table (jointure) dont les attributs sont la réunion des attributs des deux tables.

- La table ainsi obtenue est calculable à partir des deux tables de départ.
- Elle permet de retrouver les deux tables de départ.
- Elle est en général de volume moindre que le total du volumes des 2 tables.

Exemple (avec les 13 relations vues précédemment) :

1. et 8.
Employé(ENom, Salaire)
Travaille à(ENom, RNom) } Employé Travaille à(ENom, Salaire, RNom) α

2. et 9.
Rayon(RNom, RNuméro)
Est Responsable De(ENom, RNom) } Rayon Resp(RNom, RNuméro, ENom) β

Que l'on peut aussi appeler ChefRayonNom

3. et 10.
Article(ANom, ANuméro)
Présent(ANom, RNom) } ArticlePrésent(ANom, RNom, ANuméro) γ

5. et 13.
Commande(CoNuméro, Date)
Emane De(CoNuméro, CNom) } Commande Emane De(CoNuméro, Date, CNom) δ

La nouvelle BD va avoir comme tables
α, β, γ, δ, 4. , 6. , 7. , 11. , 12.

Attention : Il n'est pas intéressant d'associé 4 et 11 car il n'y a seulement inclusion des clés et pas égalité. La table obtenue par jointure peut être beaucoup plus grande que les deux tables de départ. Par exemple, pour chaque article fournit par un fournisseur il faudra donner l'adresse de celui-ci, d'où des répétitions malvenues.

La table 7. Est une projection de β, donc cette table est inutile.
On a donc finalement une BD à 8 relations seulement.

Attention :
Dans cette simplification de deux tables ayant une clé commune en une seule table il peut y avoir un problème car les projections sur la clé commune des ces 2 tables doivent être égales (il ne peut pas y avoir d'attribut vide)
Il existe deux solutions :
1) Interdire cette chose la par une contrainte.
2) Compléter les u-plets sur les attributs non définit par la valeur « null » (⊥)

5) Les 5 opérations fondamentales de l'algèbre relationnelle.

1. Une opération qui rajoute des lignes : La réunion

Si T_1 et T_2 ont les mêmes attributs on a la réunion $T_1 \cup T_2$.

2. Une opération qui rajoute des colonnes (et multiplie les lignes) : Le produit cartésien

Si T_1 et T_2 sont sans attribut commun
$T_1 \times T_2$: on prend une ligne de T_1 et une ligne de T_2 de toutes les façon possibles.

Exemple :

T_1	A_1	A_2
	a	b
	c	d
	e	f

T_2	B_1	B_2	B_3
	i	j	k
	l	m	n

$T_1 \times T_2$	A_1	A_2	B_1	B_2	B_3
	a	b	i	j	k
	a	b	l	m	n
	c	d	i	j	k
	c	d	l	m	n

On a les propriétés suivantes :

- nb colonnes($T_1 \times T_2$) = nb colonnes(T_1) + nb colonnes(T_2)
- nb lignes($T_1 \times T_2$) = nb lignes(T_1) * nb lignes(T_2)

- Si $R \subset D_1 \times D_2$ et $S \subset E_1 \times E_2 \times E_3$
  $R \times S = \{(x, y, z, t, u) \mid (x, y) \in R \text{ et } (z, t, u) \in S)\}$

- En terme de fonction, si :
  F_1 est l'ensemble de fonctions de $\{A_1, A_2\}$ qui envoient A_1 dans D_1 et A_2 dans D_2
  F_2 est l'ensemble de fonctions de $\{B_1, B_2, B_3\}$ qui envoient B_1 dans E_1, B_2 dans E_2 et B_3 dans E_3
  $F_1 \times F_2$ = les fonctions de domaine $\{A_1, A_2, B_1, B_2, B_3\}$ dont la restriction à $\{A_1, A_2\}$ est dans F_1 et dont la restriction à $\{B_1, B_2, B_3\}$ est dans F_2 .

Les attributs de $T_1 \times T_2$ sont ceux de la réunion des attributs de T_1 et de T_2.

3. Une opération qui enlève des colonnes : La projection

Soit T une table, et soient A_1, ……, A_n les attributs de la table.
Soient A_{i_1}, ……, A_{i_k} tels que $1 \leq i_1 \leq \ldots\ldots \leq i_k \leq n$ on a :
$\text{Proj}_{i_1 \ldots i_k}(T)$ =les lignes de T dont on retient que les valeurs dans les colonnes $i_1 \ldots\ldots i_k$.

Exemple :

T	A	B	C	D	E
	i	j	k	l	m
	n	o	p	q	r
	s	t	u	v	w

$\text{Proj}_{B,C}(T)$	B	C
	j	k
	o	p
	t	u

4. Une première opération qui enlève des lignes : La différence

On suppose T_1 et T_2 ont les mêmes attributs.
$T_1 \setminus T_2$ sont les lignes de T_1 qui n'apparaissent pas dans T_2.

5. Une deuxième opération qui enlève des lignes : La sélection

C'est en fait une famille d'opérations paramétrée par des formules.

Soient A_1, ……, A_n les attributs de T.

- Formules atomiques :
  $A = $B
  $A < $B
  $A = a (où a est dans le domaine associé à l'attribut A)
  $A > a
  $A< a

On considère les formules obtenues à partir des atomiques à l'aide des connecteurs $\neg$, $\wedge$, $\vee$ (négation, conjonction, disjonction)
A chaque formule ϕ on associe un opérateur de sélection $\sigma(\phi)$.
$\sigma(\phi)$ prend une table (d'attributs A_1, ……, A_n) et renvoie une sous-table (on a sélectionné certaines lignes de T)

- Construction des $\sigma(\phi)$ par récurrence sur ϕ

Cas ou ϕ est $A = $B
$\sigma($A = $B)(T)$ = les lignes de T dont les éléments des colonnes A et B sont égaux.

Cas ou ϕ est $A = a
$\sigma($A = a)(T)$ = les lignes de T dont l'élément en colonne A vaut a.

Cas ou ϕ est $A < $B
$\sigma($A < $B)(T)$ = les lignes de T dont l'élément en colonne A est plus petit que l'élément de colonne B.

Cas ou ϕ est $A < a
$\sigma($A < a)(T)$ = les lignes de T dont l'élément en colonne A est plus petit que l'élément a.

Cas ou ϕ est $A > a
$\sigma($A > a)(T)$ = les lignes de T dont l'élément en colonne A est plus grand que l'élément a.

Cas ou ϕ est $\neg$G
$\sigma(\neg G)(T) = T \setminus \sigma(G)(T)$ = les lignes de T non sélectionnées par $\sigma(G)$.

Cas ou ϕ est G$\vee$H
$\sigma(G \vee H)(T) = \sigma(G)(T) \cup \sigma(H)(T)$ = les lignes de T sélectionnées par l'un au moins de $\sigma(G)$ ou $\sigma(H)$.

Cas ou ϕ est G$\wedge$H
$\sigma(G \wedge H)(T) = \sigma(G)(T) \cap \sigma(H)(T)$ = les lignes de T sélectionnées par à la fois $\sigma(G)$ et $\sigma(H)$.

Rappel :

Les 5 opérations fondamentales sont les suivantes :
- **L'union**, qui rajoute des lignes.
- **Le produit cartésien**, qui rajoute des colonnes et des lignes
- **La différence**, qui enlève des lignes.
- **La sélection**, qui enlève des lignes.
- **La projection**, qui enlève des colonnes.

➢ La projection

Si T a n attributs $\{A_1, \ldots\ldots, A_n\}$ à chaque partie $X \subseteq \{A_1, \ldots\ldots, A_n\}$ est associé une projection :

- Cas $X=\{A_1, \ldots\ldots, A_n\}$ alors $proj_x$ est l'identité.

- Cas $X=\varnothing$ f⌈X est une fonction de domaine ∅**,** il existe une et une seule fonction de domaine vide, celle dont le graphe est vide.
  - Combien y a t-il d'ensembles de fonctions de domaine vide ?
    - Il y en a deux : ∅ et {∅} :
      ∅ est la table vide d'arité 0 (en terme de réponse à une requête, la réponse est non)
      {∅} est la table non vide d'arité 0 qui ne comporte qu'une ligne laquelle est vide (en terme de réponse à une requête, la réponse est oui)

Exemple :

Soit R(x,y)
On demande les x tels que R(x,x)
On fait une sélection : $\sigma_{\{x=y\}}(R) = \{(a,b) \in$ table de R $\mid a=b\}$
Ensuite on projette : $proj_{\{1\}}(\sigma_{\{x=y\}}(R)) = \{a \mid (a, a) \in$ table de R$\}$
On refait une projection indexé par l'ensemble vide d'attributs.
$proj_{\varnothing}(proj_{\{1\}}(\sigma_{\{x=y\}}(R)))$.

- Si $proj_{\varnothing}(proj_{\{1\}}(\sigma_{\{x=y\}}(R))) = \{[\,]\}$ singleton contenant seulement liste vide alors il existe x tel que R(x,x).
- Si $proj_{\varnothing}(proj_{\{1\}}(\sigma_{\{x=y\}}(R))) = \varnothing$ alors il n'existe pas x tel que R(x,x).

D'autres opérations de l'algèbre relationnelle

Toutes ces opérations vont être obtenues en composant les 5 opérations fondamentales.

Opérations booléennes :
L'union $\cup$ et la différence \ ont déjà été mises explicitement.
L'intersection $\cap$ est définie par $A \cap B = A \setminus (A \setminus B)$

Jointure naturelle :
Soit R d'attributs $\{A_1, \ldots\ldots, A_n, C_1, \ldots\ldots, C_p\}$ et S d'attributs $\{B_1, \ldots\ldots, B_m, C_1, \ldots\ldots, C_p\}$
$\{C_1, \ldots\ldots, C_p\}$ sont les attributs communs à R et S.
On a la jointure naturelle de R et S :

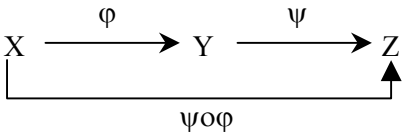
$$R \bowtie S = \{(a_1, \ldots, a_n, b_1, \ldots, b_m, c_1, \ldots, c_p) \mid (a_1, \ldots, a_n, c_1, \ldots, c_p) \in R \text{ et } (b_1, \ldots, b_m, c_1, \ldots, c_p) \in S$$

Par analogie avec les fonctions on a :

$$R \bowtie S = \{f \cup g \mid f \text{ fonction de domaine } \{A_1, \ldots, A_n, C_1, \ldots, C_p) \text{ dans } R$$
$$g \text{ fonction de domaine } \{B_1, \ldots, B_m, C_1, \ldots, C_p) \text{ dans } S$$
$$\text{telles que } f \div \{C_1, \ldots, C_p\} = g \div \{C_1, \ldots, C_p\}\}$$

Exemple :

Soit :



graphe(φ) = {(a, b) | a∈X, b∈Y φ(a)=b}
graphe(ψ) = {(b, c) | b∈Y, c∈Z ψ (b)=c}
graphe(ψoφ) = {(a, c) | a∈X, c∈Z ψ(φ(a))=c}
mais (a, c)∈graphe(ψoφ) si et seulement si il existe b∈Y tel que :
(a, b)∈graphe(φ)
(b, c)∈graphe(ψ)
c'est à dire si (a, b, c)∈(graphe(φ) ⋈ graphe(ψ))
d'ou graphe(ψoφ) =$\text{proj}_{\{X, Z\}}$(graphe(φ) ⋈ graphe(ψ))

Cas particuliers :

1. p=0
R d'attributs $\{A_1, \ldots\ldots, A_n\}$ et S d'attributs $\{B_1, \ldots\ldots, B_m\}$, il n'y a pas d'attribut commun.
Alors R⋈S coïncide avec le produit cartésien R×S.

2. m=n=0
R et S ont pour attributs $\{C_1, \ldots\ldots, C_p\}$, tous les attributs sont commun à R et S.
Alors R⋈S coïncide avec l'intersection R∩S.

- On peut obtenir la jointure naturelle à partir des 5 opérations fondamentales :

  $R \bowtie S = \text{proj}_{\{A_1,...,A_n,B_1,...,B_m,C_1,...,C_p\}}(\sigma_F(R \times S))$

  R est d'attributs $\{A_1, \ldots\ldots, A_n, C_1, \ldots\ldots, C_p\}$
  S est d'attributs $\{B_1, \ldots\ldots, B_m, C_1', \ldots\ldots, C_p'\}$ on a renommé les C_i en C_i' pour faire le produit cartésien.
  F est la formule $x_{c_1} = x_{c_1'}$, $x_{c_2} = x_{c_2'}, \ldots\ldots\ldots, x_{c_p} = x_{c_p'}$

θ-jointure

On a les θ-relation : $=, \neq, >, \geq, <, \leq$
Soient R d'attributs $\{A_1, \ldots\ldots, A_n\}$ et S d'attributs $\{B_1, \ldots\ldots, B_m\}$
$R \underset{A\theta B}{\bowtie} S = \{(a_{A_1},...,a_{A_n},b_{B_1},...,b_{B_n}) \mid a_A \theta b_B\}$ c'est une sélection dans $R \times S$

Semi-jointure $R \ltimes S$

$R \ltimes S$ est la projection de $R \bowtie S$ sur les attributs de R

Exemple :

Reprenons l'exemple de la base de données du supermarché.
On pose la requête : Quels sont les clients ayant commandé du Brie ?

Rappelons qu'on avait les tables :
Comprend(CoNuméro, ANom, Quantité)
Commande Emane De(CoNuméro, CNom)

$\text{proj}_{\{CNom\}}(\sigma(Anom = \text{'Brie'})(\text{Commande Emane De} \bowtie \text{Comprend}))$ = La liste (en colonne) des clients ayant commandé du Brie

Chapitre 3. Les calculs relationnels

Rappel :

- Logique du 1er ordre d'un langage relationnel :

On se donne un langage comprenant une famille de symboles de constantes et une famille de symboles de relations de diverses arités.

Avec cela on construit des formules atomiques
Soit R un symbole de relation d'arité k

- $R(x_1,\ldots\ldots, x_k)$ est une formule atomique si $x_1,\ldots\ldots, x_k$ sont des variables.
- On peut aussi dans une formule atomique substitué une occurrence de variable par une autre variable ou une constante.

Exemple :

Si R est ternaire R(x, y, z), R(x, z, z), R(x, a, y) sont des formules atomiques.

Formules :

- Les formules atomiques sont des formules
- Si F est une formule alors $\neg F$ est une formule.
- Si F et G sont des formules alors $F \wedge G$, $F \vee G$ et $F \Rightarrow G$ sont des formules.
- Si F est une formule et x est une variable alors $\exists xF$ et $\forall xF$ sont des formules

Toutes les formules sont obtenues de cette façon.

Occurrences liées et occurrences libres d'une variables :

- Dans une formule atomique toutes les occurrences de variables sont libres.
- Les occurrences libres de variables dans $\neg F$, $F \wedge G$, $F \vee G$, $F \Rightarrow G$ sont celles de F et G.
- Les occurrences liées de variables dans $\neg F$, $F \wedge G$, $F \vee G$, $F \Rightarrow G$ sont celles de F et G.
- Toutes les occurrences de x dans $\exists xF$ et $\forall xF$ sont liées.
- Les occurrences libres des autres variables que x dans $\exists xF$ et $\forall xF$ sont celles de F.

Sémantique des formules :

On se donne :

- Un ensemble non vide D dit domaine.
- Un élément de D pour chaque constante du langage logique .
- Une relation d'arité k pour chaque symbole de relation d'arité k du langage logique.

On a la structure du langage :
$(D, \alpha_c, \ldots, \rho_R, \ldots) = M$
$\alpha_c \in D$ pour chaque constante c.
$\rho_R \subseteq D^k$ pour chaque symbole R d'arité k.

Sémantique d'une formule dans une telle structure :
On fait la construction par récurrence sur les formules

➢ Si F comporte k variables libres $x_1,\ldots\ldots, x_k$ alors Sém(F, M)$\subseteq D^k$.

Cas particuliers :

- Cas k=0

Sém(F, M)$\subseteq D^k = \{\varnothing\}$

Si Sém(F, M)$\subseteq D^k = \{\varnothing\}$ F est vraie dans M.

Si Sém(F, M)$\subseteq D^k = \varnothing$ F est fausse dans M.

- Cas F atomique

$R(x_1,\ldots\ldots, x_k)$ alors Sém(F, M)= ρ_R.

- Cas F atomique obtenue par substitution de a à x dans G atomique

On a Sém(F, M)= $\text{proj}_{\text{les variables autres que } x_i}(\sigma(x_i = a)(\text{Sém}(G, M)))$

- Cas F atomique obtenue par substitution de x_j à x_i dans G atomique

On a Sém(F, M)= $\text{proj}_{\text{les variables autres que } x_i}(\sigma(x_i = x_j)(\text{Sém}(G, M)))$

➢ Si F est G$\vee$H

Sém(F, M) = {$(a_1,\ldots\ldots, a_k)$ | la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de G est dans Sém(G, M) **ou**
la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de H est dans Sém(H, M)}

➢ Si F est G$\wedge$H

Sém(F, M) = {$(a_1,\ldots\ldots, a_k)$ | la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de G est dans Sém(G, M) **et**
la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de H est dans Sém(H, M)}

➢ Si F est G$\Rightarrow$H

Sém(F, M) = {$(a_1,\ldots\ldots, a_k)$ | la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de G est dans Sém(G, M) **implique**
la restriction de $(a_1,\ldots\ldots, a_k)$ aux composantes correspondantes aux variables de H est dans Sém(H, M)}

➢ Si F est $\neg$G

Sém(F, M)=$D^k \setminus$ Sém(G, M)

Théorème de complétude (Gödel 1930)

F est vraie dans tous les modèles si et seulement si F est prouvable.

Calculs relationnels = de la logique adaptée aux BD relationnelles
Il y a plusieurs façons de présenter :
Calculs relationnels à variables domaine
Calculs relationnels à variables t-uplets

Problème :
Mettre ou pas l'égalité
Mettre ou pas les comparaisons $<$, $>$, $\neq$
Richesse des formules qu'on se permet.

1) Calcul relationnel purement conjonctif à variables domaine

On prend un schéma relationnel fixé R_1, ……, R_n d'une BD, chaque R_i à des attributs.
A chaque attribut A est associé un certain domaine (infini) D(A)
Nous fixons un domaine D qui contient tous les D(A)

Définition :
Les formules du calcul purement conjonctif associé au schéma relationnel et au domaine D fixé sont les formules n'utilisant que $\wedge$ et $\exists$ de la logique dont les constantes sont tous les éléments de D et dont les relations sont les R_i.

$R_i(V_1(A_1), \ldots\ldots, V_k(A_k))$ est une formule atomique si R_i a pour attributs A_1, ……, A_k et si $V_i(A_i)$ est une variable ou un élément de D.

2) Calcul relationnel purement conjonctif à variables domaine

Définition :

- On utilise des formules utilisant uniquement le connecteur $\wedge$ et la quantification $\exists$.
- On part : - du schéma relationnel d'une BD
  - de la BD elle-même (c'est à dire de son contenu)
- On considère un domaine D qui contient les domaines des attributs de la BD
- On a les formules atomiques (définies par le schéma relationnel de la BD) utilisant des constantes prises dans D.
- On fait des conjonctions et des $\exists$.

Sémantique de telles formules (cas particulier de la définition générale):

- Sém($R(x_1, \ldots\ldots, x_n)$, D) = la relation associé à R dans l'état de la BD
- Sém($R(v(x_1), \ldots\ldots, v(x_n))$, D) = $\text{proj}_{\text{lesvariablesdans}\{v(x_1),\ldots,v(x_n)\}}(\sigma_\varphi(R))$

  $v(x_i)$ est une variable ou un élément de D

  φ est la conjonction des $\$i=\$j$ si $v(x_i)= v(x_j)$ $\$i=a$ si $v(x_i)=a$

  Exemple : Sém(R(x, x, a), D) = $\text{proj}_{\{x\}}(\sigma_{\$1=\$2 \wedge \$3=a}(R))$

- Sém($F \wedge G$, D) = Sém(F, D) $\bowtie$ Sém(G, D) car

  Sém($F(x_1, \ldots\ldots, x_k, z_1, \ldots\ldots, z_p) \wedge G(y_1, \ldots\ldots, y_l, z_1, \ldots\ldots, z_p)$, D)

  = $\{(x_1, \ldots\ldots, x_k, y_1, \ldots\ldots, y_l, z_1, \ldots\ldots, z_p) \mid (x_1, \ldots\ldots, x_k, z_1, \ldots\ldots, z_p) \in$ Sém(F, D) et $(y_1, \ldots\ldots, y_l, z_1, \ldots\ldots, z_p) \in$ Sém(G, D)$\}$

  = Sém(F, D) $\bowtie$ Sém(G, D)

Notation :

- Etant donné l'état I de la BD on note le domaine actif :
  adom(I) = les constantes figurant dans les tables
- adom(F) = les constantes figurant dans F.

Théorème d'indépendance de la sémantique (des formules purement conjonctive) vis à vis du domaine :

Si $F(x_1, \ldots\ldots, x_k)$ est purement conjonctive et si $s : \{x_1, \ldots\ldots, x_k\} \to D$ est une assignation de valeurs telle que $(s(x_1), \ldots\ldots, s(x_k)) \in$ Sém(F, D) alors tous les éléments $s(x_1), \ldots\ldots, s(x_k)$ sont dans adom(I) $\cup$ adom(F).

En d'autre termes, la domaine D n'intervient pas réellement ce qui est très rassurant puisque ce domaine D est arbitraire.

Preuve (Par récurrence):

- Cas F atomique $R(x_1, \ldots\ldots, x_k)$
  La sémantique est la valeur de la table associé à R donc tous les éléments sont dans adom(I).

- Cas F atomique $R(v(x_1), \ldots\ldots, v(x_k))$
  Ou $v(x_i)$ est une variable ou un élément de D. La sémantique s'obtient par la projection d'une sélection de la sémantique de $R(x_1, \ldots\ldots, x_k)$. Laquelle sélection ne fait intervenir que des constantes de adom(F) donc tous les éléments sont dans adom(F).

- Cas $F \wedge G$
  Les éléments des uplets de Sém(F, D) $\bowtie$ Sém(G, D) sont les uplets de Sém(F, D) ou de Sém(G, D).
  Donc tous les éléments sont dans adom(F).

- Cas $\exists x F$
  Ce cas est évident puisque $Sém(\exists x\ F, D) = proj_{\text{les variables autres que } x}(Sém(F, D))$

Définition :
Une formule purement conjonctive est sous forme normale si elle est de la forme :
$\exists\ y_1, \ldots\ldots, y_l\ F(x_1, \ldots\ldots, x_l, y_1, \ldots\ldots, y_l)$ où F est une conjonction sans quantification.

Fait :
Toute formule purement conjonctive équivaut à une formule en forme normale.
« équivaut » veut dire « avoir la même sémantique ».

Preuve :
Soit $\exists \vec{y}\ F(\vec{y}, \vec{x}, \vec{t}) \wedge \exists \vec{z}\ G(\vec{z}, \vec{u}, \vec{t})$

1. On renomme les variables $\vec{z}$ en $\vec{z}'$ de sorte qu'elles soient différentes de variables $\vec{y}$.
2. On regroupe les $\exists$ en $\exists \vec{y} \exists \vec{z}'\ (F(\vec{y}, \vec{x}, \vec{t}) \wedge G(\vec{z}', \vec{u}, \vec{t}))$

Exemple :

$\exists y_1\ \exists y_2\ F(y_1, y_2, x, t) \wedge \exists y_1\ \exists z\ G(y_1, z, u_1, u_2, t)$ on renomme :
$\exists y_1\ \exists y_2\ F(y_1, y_2, x, t) \wedge \exists y_1'\ \exists z\ G(y_1', z, u_1, u_2, t)$ on obtient donc :
$\exists y_1\ \exists y_2\ \exists y_1'\ \exists z\ (F(y_1, y_2, x, t) \wedge G(y_1', z, u_1, u_2, t))$

Exemple (avec une BD de cinéma) :

Film(Titre, Réalisateur, Acteur) table à 3 attributs de domaine les chaînes de caractères
Salle(Cinéma, Adresse, Téléphone)
Programme(Cinéma, Titre, Horaire)

On a les requêtes suivantes :
1. Qui est le réalisateur de « Cris et chuchotements »
2. Dans quel cinéma passe « Cris et chuchotements »
3. Quel est l'adresse et le téléphone du St André des Arts
4. Donner les noms et adresses des cinémas passant un film de Bergman
5. Y a t-il un film de Bergman à l'affiche ?

Suite page suivante……

Les réponses sont les suivantes :

1. $\exists Y$ Film(« Cris et chuchotements », X, Y)
2. $\exists H$ Programme(X, « Cris et chuchotements », H)
3. Salle(« St André des Arts », X, Y)
4. $\exists X\ \exists Y\ \exists H\ \exists T$ (Film(X, « Bergman », Y) $\wedge$ Programme(C, X, H) $\wedge$ Salle(C, A, T))
5. $\exists X\ \exists Y\ \exists H\ \exists C$ (Film(X, « Bergman », Y) $\wedge$ Programme(C, X, H))

Revenons sur la requête numéro 5. La relation naturelle pour 5 est la suivante :
Quels films de Bergman passent où et quand ? On a la requête :
(Film(X, « Bergman », Y) $\wedge$ Programme(C , X, H))
titre acteur salle horaire

C'est une relation en X, Y, C, H, il y a deux cas possibles :

1. Cette relation n'est pas vide (si on oublie pour chaque ligne de cette relation les 4 éléments de la ligne, on obtient la liste vide) alors la réponse à la requête 5 est **OUI**.
   Ce qui veut dire que :
   $\text{proj}_{\varnothing \text{ projection sur l'ensemble vide de colonnes}}$(de cette relation) = {[ ]}

2. Cette relation est vide alors la réponse à la requête 5 est **NON**
   Ce qui veut dire que :
   $\text{proj}_{\varnothing \text{ projection sur l'ensemble vide de colonnes}}$(de cette relation) = $\varnothing$

3) Calcul relationnel purement conjonctif et algèbre relationnelle.

Pour l'algèbre relationnelle on a les opérateurs $\cup$, $\times$, proj, diff, sélection.

Définition :
Algèbre Relationnelle S.P.C. (Sélection, Projection, Produit cartésien) on compose ces 3 opérations appliquées aux relations associés aux tables de l'état de la BD et aux relations unaires singleton {(a)} où a $\in$ D.
Pour la sélection on se restreindra à $x_i = x_j$ identifier deux arguments et $x_i = a$ fixer un argument.

Définition :
Une expression sous forme normale est une expression sous la forme :
$\text{proj}_{\{j_1,\ldots,j_m\}}(\{a_1\}\times\ldots\times\{a_p\}\times\sigma_1 \circ \sigma_2 \circ \ldots \circ \sigma_i(R_1\times\ldots\times R_j))$ ou $\{a_1\}\in D \ldots \{a_p\}\in D$

Proposition :
Toute expression peut se mettre sous forme normale.

Preuve :

1. On met les proj tout au début car :
   - $\sigma_{\$i_p=\$i_q}(\text{proj}_{\{i_1,\ldots,i_k\}}(S)) = \text{proj}_{\{i_1,\ldots,i_k\}}(\sigma_{\$i_p=\$i_q}(S))$

   Exemple :

   Soit S = (a, b, c, d) des quadruplets
   $\text{proj}_{\{1,2,3\}}(S)$ = les (a, b, c) tels que $\exists$d (a, b, c, d)$\in$S
   $\sigma_{\$1=\$2}(\text{proj}(S))$ = les (a, a, c) tels que $\exists$d (a, a, c, d)$\in$S
   = $\text{proj}_{\{1,2,3\}}$(l'ensembles des quadruplets (a, a, c, d) de S)
   = $\text{proj}_{\{1,2,3\}}(\sigma_{\$1=\$2}(S))$

   - $\sigma_{\$i_p=a}(\text{proj}_{\{i_1,\ldots,i_k\}}(S)) = \text{proj}_{\{i_1,\ldots,i_k\}}(\sigma_{\$i_p=a}(S))$
   - $S\times \text{proj}_{\{i_1,\ldots,i_k\}}(T) = \text{proj}_{\{1,2,\ldots,m,m+i_1,\ldots,m+i_k\}}(S\times T)$ ou S est d'arité m

2. $\text{proj}_I(\text{proj}_J(S)) = \text{proj}_I(S)$ ou $I \subseteq J$

Avec 1. et 2. On met toutes les projections en tête et on les réduit à une seule.

3. Après la proj initiale on ne trouve que des sélections et des produits.
   On peut ramener les sélections avant les produits car :
   $S\times\sigma_\varphi(T) = \sigma_\psi(S\times T)$ et $\sigma_\varphi(S)\times T = \sigma_\varphi(S\times T)$ où ψ s'obtient en changeant dans φ les \$i par \$i+n si n est l'arité de S.

4. On sort les singletons car $\sigma(\{a\}\times S) = \{a\}\times\sigma(S)$ car :
   $\sigma_\varphi(S)\times\{a\} = \sigma_\varphi(S\times a)$ et $\{a\}\times\sigma_\varphi(S) = \sigma_\psi(\{a\}\times S)$ où ψ s'obtient en changeant dans φ les \$i par \$i+1.

Théorème d'équivalence :
Les expressions relationnelles définissent les mêmes relations que les requêtes purement conjonctives.

Remarques :

- Une expression relationnelle peut être toujours vide (on dit « non satisfaisable ») par exemple : $\sigma_{\$x=a}(\sigma_{\$x=b}(R))$ où a et b sont différents, constantes de adom(I).
- En revanche, une requête purement conjonctive qui ne fait intervenir que des constantes des tables est toujours satisfaite. Pour avoir une relation vide il faut utiliser des constantes hors de la table. Film(X, 1999, Y) est vide car 1999 n'est jamais le nom du réalisateur quelque soit l'état de la BD.

Cours de Base de données

13.03.2000

4) Calcul relationnel purement conjonctif à variables domaine

Rappel :

- On utilise des formules utilisant uniquement la conjonction $\wedge$ et la quantification $\exists$.
- On considère un domaine D qui contient les domaines des attributs de la BD
- On a les formules atomiques ($R(v(x_1), \ldots\ldots, v(x_n))$ où $v(x_i)$ est une variable ou une constante prise dans D.
- On utilise également les formules $x = a$ où a est un élément de D

Remarque :
$x=b \wedge x=a$ n'est jamais satisfaisable (alors que sans ces formules atomiques il y a toujours un état de la BD qui rend une formule conjonctive satisfaisable).

2) Calcul relationnel purement conjonctif et algèbre relationnelle.
Algèbre SCP

On utilise les 3 opérations suivantes :

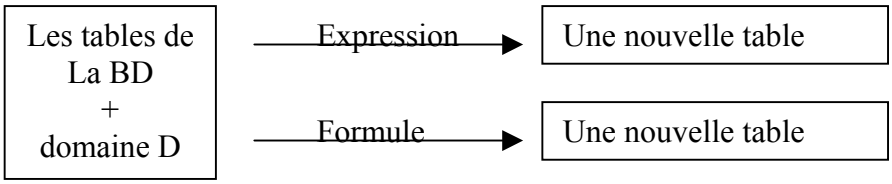
La sélection $i=$j, $i=a
La projection
Le produit cartésien

} à partir des tables de la BD et des relation unaires {(a)} pour $a \in D$

On a vu que :

1. Toute formule purement conjonctive équivaut à une formule
   $\exists x_1, \ldots\ldots, x_k$ (conjonction de formules atomiques et de formules x=a)
2. Toute expression de SCP équivaut à une expression :
   $proj_X(\{a_1\} \times ..... \times \{a_k\} \times (\sigma_1 \, o \, \sigma_2 \, o \, ...\sigma_1)(R_1 \times ..... \times R_m))$

Théorème d'équivalence :
A toute formule purement conjonctive on peut associer une expression SCP qui à la même sémantique.
Réciproquement à toute expression SCP on peut associer une formule purement conjonctive ayant la même sémantique (à une duplication près d'arguments).



Preuve :

1er partie

- Cas d'une formule atomique ($R(v(x_1), \ldots\ldots, v(x_n))$ où $v(x_i)$ est une variable ou une constante prise dans D.
  L'expression associée est la suivante :
  $\text{proj}_{\text{variables de } R(v(x_1)\ldots v(x_n))}(\sigma_{\$i=\$j,\ldots,\$i=a \text{ si } v(x_i)=v(x_j) \text{ } v(x_i)=a})(R)$

  Exemple : Si on a R(x, x, a, y) on fait $\text{proj}_{1,4}(\sigma_{\$1=\$2,\ \$3=a})(R)$

  A la formule x=a on associe l'expression {(a)}

- Cas d'une formule $F \wedge G$ si à F et G on sait déjà associer E_F et E_G.
  $F(x_1, \ldots\ldots, x_n, z_1, \ldots\ldots, z_p) \wedge G(y_1, \ldots\ldots, y_m, z_1, \ldots\ldots, z_p)$
  L'expression associée est la suivante :
  $\text{proj}_{\{1,\ldots,n,\ n+1,\ldots,\ n+p,\ n+p+1,\ldots,\ n+p+m\}}(\sigma_{\$n+i=\$n+p+m+i})(E_F \times E_G)$ $E_F \times E_G$ est d'arité n+p+m+p

- Cas d'une formule $\exists x\ F(x, x_1, \ldots\ldots, x_n)$ si à F on sait associer E_F,
  on associe à $\exists x F$ l'expression
  $\text{proj}_{\{2,\ldots,\ n+1\}}(E_F)$

2eme partie

- A {(a)} on associe la formule x=a

- A R on associe la formule $R(x_1, \ldots\ldots, x_n)$

- Soit E
  - On considère $\text{proj}_X(E)$, si on sait déjà associer une formule F_E à E on associe à $\text{proj}_X(E)$ la formule $\exists x_1, \ldots\ldots, x_k\ F_E$ où les x_i correspondent aux variables de F, c'est à dire aux positions dans la relation E qui sont hors de X.
    Exemple : Si S à 3 arguments F(x, y, z)
    $\text{Proj}_{\{2,3\}}(S) \rightarrow \exists x\ F(x, y, z)$
  - Si $\sigma_{\$i=\$j}(E)$ à E on associe $F(x_1, \ldots\ldots, x_n)$
  - Si $\sigma_{\$i=\$j}(E)$ à E on associe F(où on remplace x_j par x_i)
    La relation associée à F comporte donc un argument de moins, d'où la nécessité de dupliquer cet argument)
  - Si $\sigma_{\$i=a}(E)$ à E on associe F(où on remplace x_i par a)
    Exemple : Si S à 3 arguments F(x, y, z)
    $\sigma_{\$1=\$3}(S) \rightarrow F(x, y, x)$

- Si à E_1 et E_2 on sait associer $F_1(x_1, \ldots\ldots, x_n)$ et $F_2(y_1, \ldots\ldots, x_p)$ à $E_1 \times E_2$ on associe :
  $F_1(x_1, \ldots\ldots, x_n) \wedge F_2(y_1', \ldots\ldots, x_p')$
  On renomme les variables pour qu'elles soient différentes des

Exemples :

Reprenons la BD de cinéma suivante :
Film(titre, réalisateur, acteur)
Salle(cinéma, adresse, téléphone)
Programme(cinéma, titre, horaire)

- Quelles sont les paires d'acteurs ayant chacun dirigé l'autre ?
  Formule purement conjonctive : $\exists t_1 \exists t_2(\text{Film}(t_1, x, y) \wedge \text{Film}(t_2, y, x))$
  Algèbre relationnelle : $\text{proj}_{\{2, 3\}} (\sigma_{\$2=\$6, \$3=\$5}(\text{Film} \times \text{Film}))$

- Quels sont les acteurs qui ont joué dans un film qu'ils ont réalisé eux-mêmes ?
  Formule purement conjonctive : $\exists t \text{ Film}(t, x, x)$
  Algèbre relationnelle : $\text{proj}_{\{2\}} (\sigma_{\$2=\$3}(\text{Film}))$

- Quels sont les acteurs ayant joué ensemble ?
  Formule purement conjonctive : $\exists t \exists z (\text{Film}(t, z, x) \times \text{Film}(t, z, y))$
  Algèbre relationnelle : $\text{proj}_{\{3, 6\}} (\sigma_{\$1=\$4, \$2=\$5} (\text{Film} \times \text{Film}))$

- Donner la réponse (« Apocalypse Now », « Coppola ») quel que soit l'état de la BD
  Formule purement conjonctive : x = « Apocalypse Now » $\wedge$ y = « Coppola »)
  Algèbre relationnelle : {(« Apocalypse Now »)} × {(« Coppola »)}

3) Calcul relationnel conjonctif avec comparaisons et égalité

On rajoute les formules atomiques suivantes :
- x=y, x<y, x>y, x<=y, x≠y
- x=a x<a, x>a, x>=a, x≠a

Attention :
Le théorème d'indépendance vis à vis du domaine (qui était vrai avec le calcul conjonctif, même avec les x=a) devient faux.

Exemple :
x=y a pour solution les (d, d) avec d dans D, or d peut ne pas être dans le domaine actif de la BD
adom(I) = constantes de l'état actuel I de la BD.
Idem pour x<y, x>y, x<=y, x≠y, x<a, x>a, x>=a, x≠a. Il y a des solutions d dans D qui peuvent être différentes de a et différentes des constantes de la BD.

Solution à ce problème :

Définition :
1) Une variable x est saine dans la formule conjonctive F si
   - elle figure dans un prédicat de base R (comme l'un des arguments)
   - elle figure dans une égalité x=a
   - elle figure dans une égalité x=y et on sait déjà que y est saine (ça veut dire que la formule comprend une chaîne d'égalités $x_{i_1} = x_{i_2}$ et ..... et $x_{i_{n-1}} = x_{i_n}$ où x est x_{i_1} et x_{i_n} figure dans un prédicat de base R ou bien dans une égalité $x_{i_n} = a$
2) Une formule conjonctive est saine si toutes ses variable sont saines

Théorème :
La sémantique des formules saines est indépendante du domaine D choisi.

Théorème d'équivalence :
Les formules saines conjonctives avec comparaisons et les expressions de SCP dans lesquelles on s'autorise des sélections par comparaisons donnent des sémantiques qui se correspondent.

Preuve :

1er partie

- $\sigma_{\$i<\$j}(S)$ si S se traduit par la formule F alors
  $\sigma_{\$i<\$j}$ se traduit par $F_1(x_1, \ldots\ldots, x_k) \wedge x_i<x_j$ comme F est saine
- idem pour $\sigma_{\$i<a}$, $\sigma_{\$i>a}$, $\sigma_{\$i\neq a}$, $\sigma_{\$i=j}$.

2eme partie

- Réciproquement, si F correspond à E alors
  $F \wedge x_i<x_j$ correspond à $\sigma_{\$i<\$j}(E)$

Exemples :

Reprenons la BD de cinéma suivante :
Film(titre, réalisateur, acteur)
Salle(cinéma, adresse, téléphone)
Programme(cinéma, titre, horaire)

- Quels sont les acteurs ayant joué dans au moins 2 films de Bergman ?
  Formule purement conjonctive : $\exists T \exists U$ (Film(T, Bergman, X) $\wedge$ Film(U, Bergman, X) $\wedge$ T≠U)
  Algèbre relationnel : $proj_{\{3\}}$ ($\sigma_{\$2=\$5=\text{"Bergman"}, \$3=\$6, \$1\neq\$4}$ (Film × Film))

- Quels acteurs ont joué dans un film non dirigé par Woody Allen ?
  Formule purement conjonctive : $\exists T \exists U$ (Film(T, U, X) $\wedge$ U≠ « Woody Allen »)
  Algèbre relationnel : $proj_{\{3\}}$ ($\sigma_{\$2 \neq \text{« Woody Allen »}}$(Film)

Reprenons maintenant la BD du supermarché :
Comprend(CoNuméro, ANom, Quantité)
Ordre Emane De(CoNuméro, Date, CNom)

- Quels sont les clients ayant commandé au moins un article en plus de 10 exemplaires ?
  $\exists T \exists U \exists V \exists W$ (Ordre Emane De(T, U, X) $\wedge$ Comprend(T, V, W) et W>10)
  $proj_{\{3\}}$ ($\sigma_{\$1=\$4, \$6>10}$(Ordre Emane De × Comprend))

4) Calcul relationnel à variables domaine le plus général.

On s'autorise les comparaisons comme les formules atomiques.
On s'autorise également les négations, disjonctions, et quantifications universelles.

Hélas, les catastrophes pullulent (pas d'indépendance vis à vis du domaine) :

1. Problème de disjonction :
   - $p(a, a) \vee p(x, x)$ si (a,a) vérifie p alors tout x de D est solution de cette requète.
   - $\exists y\ (p(x, y) \vee p(y, z))$ avec $p=\{(a, b)\ (c, d)\}$, $q=\{(e, f)\}$ et $D=\{a, b, c, d, e, f, g\}$ ou g est une constante hors de la tables de la BD. Donc le couple (a, g) satisfait la requète
2. Problème avec $\forall$:
   - $\forall y\ p(x, y, y)$ si $p=E\times E\times E$ alors si D=E les solutions sont les éléments de E, mais si $D \grave{U} E$ il n'y a aucune solution.

Définitions:

1) La variable x est saine dans une conjonction $F_1 \wedge \ldots\ldots \wedge F_k$ si :
   - elle figure dans l'une des formules F_i qui n'est pas une négation ni une comparaison.
   - elle figure dans l'une des formules F_i qui est une égalité x=a.
   - elle figure dans l'une des formules F_i qui est une égalité x=y où y a déjà été reconnue saine dans cette conjonction.
2) Une formule est saine si :
   - elle n'utilise pas $\forall$ (Rappel : $(\forall x\ P(x)) \Leftrightarrow (\neg\exists x\ \neg P(x))$)
   - toute sous formule de la forme $G \vee H$ est telle que toute les formules G et H ont exactement les mêmes variables libres.
   - Toute sous formule qui est une conjonction maximale a toutes ses variables libres saines (cette clause s'applique en particulier aux formules atomiques qui ne sont pas conjointes dans la grande formule)
   - Une négation ne peut apparaître que dans l'un des composants d'une conjonction maximale comprenant au moins une formule non niée qui n'est pas une comparaison.

Théorème :
La sémantique des formules saines est indépendante du domaine D choisi.

4) Calcul relationnel à variables domaine le plus général.

Théorème d'équivalence :

1. A toute expression du calcul de l'algèbre relationnelle on peut associer une formule saine de même sémantique.
2. A toute formule saine on peut associer une expression de l'algèbre relationnelle.

Preuve:

1)
   - Le produit cartésien se traduit par la conjonction.
   - La projection se traduit par le $\exists$
     $proj_X$ se traduit par $\exists y_1 \ldots\ldots \exists y_k\ (F_r)$ où les y_i sont les variables correspondant aux attributs non dans X.
   - La sélection se traduit en conjoignant avec des comparaisons
   - La réunion $R \cup S$ se traduit par la disjonction: $Fr \wedge Fs$ traduit $R \cup S$ si Fr et Fs traduisent R et S
     Comme R et S ont les mêmes attributs, les formules $Fr \wedge Fs$ ont les mêmes variables libres, et donc leur disjonction est saine.
   - La différence $R \backslash S$ se traduit par $Fr \wedge \neg Fs$
     R et S ont les mêmes attributs donc Fr et Fs ont les mêmes variables libres, donc $Fr \wedge \neg Fs$ est une formule saine.
2) Récurrence sur la formule:
   - Le cas atomique à déjà été vu avec le calcul conjonctif
   - Cas d'une disjonction:
     On peut utiliser la réunion car si F et G ont les mêmes variables libres alors les expressions associés E_F, E_G ont les mêmes attributs.
   - Cas d'une quantification $\exists$:
     On utilise la projection
   - Cas d'une conjonction maximale:
     De la forme $F_1 \wedge \ldots\ldots \wedge F_n \wedge \neg G_1 \wedge \ldots\ldots \wedge \neg G_p \wedge$comparaisons
     On sait que $n \geq 1$. Toutes les variables libres figurent dans au moins l'un des Fi, on suppose déjà associés des relations R_{Fi}, R_{Gj} aux F_i et G_j.
     1. On fait la jointure des R_{Fi} : $R_{F_1} \bowtie R_{F_2} \bowtie \ldots\ldots \bowtie R_{F_n}$
     2. On sélectionne dans cet jointure pour traduire les comparaisons.
     3. On observe que les variables libres des G_j sont ou bien fixées à une certaine valeur par les comparaisons ou bien (modulo des égalités des comparaisons) figurent parmi les variables libres de F_i.
        On se ramène donc au cas où toutes les variables des G_j figurent parmi celles de F_i.
        Donc les attributs des R_{Gj} sont tous des attributs de la jointure des R_{Fi}.
        A chaque attribut X de l'un des F_i on associe une expression E_x : si X est le k-ième argument de F_i alors $E_x = proj_{\{k\}}(R_{Fi})$
        Pour chaque j on considère le produit cartésien de R_{Gj} avec tous les E_x ou X est une variable libre des F_i qui ne figure pas dans G_j, ce produit R'_{Gj} a alors exactement les mêmes attributs que la jointure des R_{Fi}.
     4. On fait alors les différences successives de la sélection de la jointure des R_{Fi} avec R'_{G1} puis R'_{G2} …… puis R'_{Gp}.

On obtient l'expression désirée.

Remarque:
Que faire avec les ∀:
On transforme ∀x F en ¬∃x ¬F
Pour qu'une sous formule ∀x F d'une grande formule laisse cette formule saine il faut que :

1. ∀x F apparaisse au sein d'une conjonction faisant intervenir au moins un terme non nié et qui ne soit pas une comparaison.
2. ¬F doit (à équivalence logique près) être une conjonction avec au moins un terme non nié et qui ne soit pas une comparaison. Donc (à équivalence près) F doit être une disjonction dont au moins un terme nié n'est pas une comparaison.

Exemple:

On considère la BD de cinéma:
Film(titre, réalisateur, acteur)
Salle(cinéma, adresse, téléphone)
Programme(cinéma, titre, horaire)
On notera FR pour formule relationnelle et AR pour algèbre relationnelle.

- Ou peut-on voir "Annie Hall" ou "Manhattan" ?
  FR : ∃H (Programme(X, "Annie Hall", H) ou Programme(X, "Manhattan", H))
  AR : $proj_{\{1\}}$ ($\sigma_{\$2="Annie\ Hall"}$(Programme)∪ $\sigma_{\$2="Manhattan"}$(Programme))

- Quels sont les films réalisés par Woody Allen ou dans lesquels il a joué ?
  FR : ∃A Film(T, " Woody Allen ", A) ou ∃D Film(T,D, "Woody Allen")
  AR : $proj_{\{1\}}(\sigma_{\$2="Woody\ Allen"}(Film))\ \cup\ proj_{\{1\}}(\sigma_{\$3="Woody\ Allen"}(Film))$
  Regardons maintenant la formule suivante:
  ∃D ∃A (Film(T, " Woody Allen ", A) ou Film(T,D, "Woody Allen"))
  Les variables libres sont respectivement T, A et T, D. Donc bien que cette formule soit équivalente à la formule saine ci dessus cette formule n'est pas saine.

- Films réalisés par Woody Allen dans lesquels il n'a pas joué ?
  FR : ∃A (Film(T, " Woody Allen ", A) et non ∃D Film(T, "Woody Allen", "Woody Allen"))
  AR : $proj_{\{1\}}$($proj_{\{1,3\}}(\sigma_{\$2="Woody\ Allen"}(Film))\ \setminus\ proj_{\{1\}}(\sigma_{\$2=\$3="Woody\ Allen"}(Film))\times proj_{\{3\}}(Film)$))

- Films dont tous les acteurs ont tourné sous la direction de Hitchcock
  1er essai:
  FR : ∃A (Film(X, D, A) ⇒ ∃X Film(X, " Hitchcock ", A))
  Problème: "tout" titre T est solution (car faux ⇒ vrai)
  Implicite dans la requête: le film existe dans la BD.
  2eme essai:
  FR : ∃D (∃A (Film(T, D, A) et ∀A' (Film(T, D, A') ⇒ ∃X Film(X, " Hitchcock ", A')))
  Ici T est borné ,il faut que ce soit un élément de la table, car T est un élément positif alors qu'avant il était négatif.
  Par contre la formule n'est pas saine à cause du ∀, la formule saine correspondante est la suivante:
  FR : ∃D (∃A (Film(T, D, A) et non ∃A' (Film(T, D, A') et non ∃X Film(X, " Hitchcock ", A')))

  Cette formule est saine car A' apparaît bien comme terme non nié

  Cette formule est saine car T, D apparaissent bien comme terme non niés

  AR : $proj_{\{1\}}$($proj_{\{1,2\}}$(Film) \ $proj_{\{1,2\}}$(Film \ $proj_{\{2,3\}}(\sigma_{\$2="Hitchcock"}(Film))\times proj_{\{1\}}(Film)$))

Chapitre 4. Dépendances fonctionnelles et formes normales

Soient: R une table
X, Y un ensemble d'attributs
La dépendance fonctionnelle X→Y signifie que si 2 lignes de la table coïncident sur les éléments des colonnes de X alors elles coïncident aussi sur les éléments des colonnes de Y.

Exemple:
FournisseurNom → FournisseurAdresse
FournisseurNom, Article → Prix
Ville, Rue → Code Postal
Code Postal → Ville

Une dépendance fonctionnelle est un choix énoncé par le concepteur de la BD. C'est donc une contrainte que vérifie le SGBD pour tout les tables de la BD.

Cours de Base de données

27.03.2000

Chapitre 4. Dépendances fonctionnelles et formes normales

Soient: X, Y des ensembles d'attributs
La dépendance fonctionnelle $X \rightarrow Y$ signifie que si 2 lignes de la table coïncident sur tous les attributs de X alors elles coïncident aussi sur tous les attributs de Y.

Dans la conception d'une BD on fixe un ensemble F de dépendances fonctionnelles.
Question naturelle:
- quelles sont les dépendances fonctionnelles conséquences des dépendances dans F ?

Axiomes d'Armstrong des dépendances fonctionnelles:
1. Axiomes de réflexivité:
   $X \rightarrow Y$ si $X \supseteq Y$
2. Règle d'accroissement:
   De l'hypothèse $X \rightarrow Y$ on déduit la conclusion
   $XZ \rightarrow YZ$ pour tout ensemble Z
3. Règle de transitivité:
   Des hypothèses $X \rightarrow Y$ et $Y \rightarrow Z$ on déduit $X \rightarrow Z$

Notation:
$XZ = X \cup Z$
$A_1 \ldots A_n = \{$

Définition:
1) On note F^+ l'ensemble des conséquences de F par les axiomes d'Armstrong: c'est à dire le plus petit ensemble de dépendances contenant F et les axiomes de réflexivité, et stable pour les règles 2. et 3.
2) Si X est un ensemble d'attributs, on note X+ l'ensemble des attributs A tels que:
   $X \rightarrow A$ soit dans F^+

Proposition:
Les axiomes et règles suivantes se déduisent des axiomes d'Armstrong:
1) Réunion: De $X \rightarrow Y$ et $X \rightarrow Z$ on déduit $X \rightarrow YZ$
2) Pseudo transitivité: Si $X \rightarrow Y$ et $WY \rightarrow Z$ on déduit $WX \rightarrow Z$
3) Décomposition: De $X \rightarrow Y$ on déduit $X \rightarrow Z$ pour tout $Z \subseteq Y$

Preuve:
1) On a $X \rightarrow Y$ et $X \rightarrow Z$, par accroissement on en déduit:
   - $XX \rightarrow XY$ mais $XX = X$
   - $XY \rightarrow ZY$

   Par transitivité on en déduit $X \rightarrow ZY$
2) De $X \rightarrow Y$ on déduit $WX \rightarrow WY$ si on a aussi $WY \rightarrow Z$ on en déduit par transitivité $WX \rightarrow Z$
3) On a $Y \rightarrow Z$ par réflexivité (si $Z \subseteq Y$) de $X \rightarrow Y$ on déduit alors par transitivité $X \rightarrow Z$

Théorème:

1) (Cohérence des axiomes d'Armstrong):
   Si une relation satisfait les dépendances de F alors elle satisfait aussi les dépendances de F^+.
2) (Complétude des axiomes d'Armstrong):
   Si une dépendance $X \rightarrow Y$ est vraie dans toutes les relations qui satisfont toutes les dépendances de F alors $X \rightarrow F \in F^+$.

Preuve:

1) La démonstration est triviale.
2) On montre que si $X \rightarrow Y \notin F^+$ alors il y a une relation qui satisfait les dépendances de F mais pas la dépendance $X \rightarrow Y$.
   On suppose que $X \rightarrow Y \notin F^+$.
   Soit R la relation (= table) avec deux lignes:
   Ligne 1: tous les attributs valent 1
   Ligne 2: les attributs de X^+ valent 1, les autres valent 0. Il est clair que R ne satisfait pas $X \rightarrow Y$ (ni même $X^+ \rightarrow Y$)
   Montrons que R vérifie toutes les dépendances $Z \rightarrow T$ de F
   - 1^{er} cas $Z \not\subseteq X^+$
     Alors les lignes 1 et 2 ne coïncident pas sur Z, la dépendance $Z \rightarrow T$ est trivialement vérifiée.
   - 2^{eme} cas $T \subseteq X^+$
     Alors les lignes 1 et 2 coïncident sur T et la dépendance est trivialement vérifiée.
   - 3^{eme} cas $Z \subseteq X^+$
     Alors $X \rightarrow Z$ se déduit des axiomes d'armstrong
     Comme $Z \rightarrow T \in F$ (par hypothèse) on en déduit par transitivité que $X \rightarrow T$ est dans F^+, d'ou $T \subseteq X^+$ par définition de X^+, et donc on est ramené au 2^{eme} cas.

Remarque:
Le calcul de F^+ est coûteux car F^+ est de taille exponentielle en le nombre des attributs (à cause de l'axiome 1.)
En revanche, le calcul de X^+ est raisonnable:
$X_0=X$
$X_{i+1}=X_i \cup \{$les A tels qu'il existe $Y \rightarrow Z$ dans F pour laquelle $Y \subseteq X_i$ et $A \in Z\}$
$X_0 \subseteq X_1 \subseteq X_2 \subseteq \ldots\ldots$ cette suite est de la forme:
$X_0 \neq X_1 \neq X_2 \neq \ldots\ldots \neq X_m = X_{m+1} = X_{m+2} = \ldots\ldots$

Exemple:

Soit F={$_1$AB→C $_5$D→EG
$_2$C→A $_6$BE→C
$_3$BC→D $_7$CG→BD
$_4$D→B $_8$CE→AG}

On regarde X=BD

X_0=BD	
X_1=BDEG	grâce à 5
X_2=BDEGC	grâce à 6
X_3=BDEGCA	grâce à 2 ou 8
$X_4 = X_3$.	

Recouvrement minimum des dépendances
On peut avoir $F \neq G$ et $F^+ = G^+$.

Remarque:
$\{X \rightarrow A \mid A \text{ attribut et } X \rightarrow A \in F^+\}^+ = F^+$
$\{X \rightarrow A \mid A \text{ attribut et il existe Y tel que } X \rightarrow A \in F^+\}^+ = F^+$

Définition:
Ensemble minimal de dépendances:
1. Il est formé de dépendance $X \rightarrow A$ ou A est un attribut
2. Si $X \rightarrow A \in F$ alors $X \rightarrow A \notin (F \setminus \{X \rightarrow A\})^+$
3. Si $X \rightarrow A \in F$ et $X' \varnothing X$ alors $X \rightarrow A \notin ((F \setminus \{X \rightarrow A\}) \cup \{X' \rightarrow A\})^+$

Proposition:
Pour tout ensemble de dépendance F il existe F' minimal tel que $F'^+ = F^+$

Exemple:

Soit F={ ${}_1$AB→C ${}_5$D→EG
${}_2$C→A ${}_6$BE→C
${}_3$BC→D ${}_7$CG→BD
${}_4$D→B ${}_8$CE→AG}

On atomise les membres droits:
AB→C
C→A
BC→D
${}_3$ACD→B simplifié en CD→B car on a C→A
D→E
D→G
BE→C
~~${}_2$CG→B~~ on a CG→D C→A ACD→B donc on oublie CG→B
CG→D
~~CE→A~~ on l'oublie
${}_1$CE→G

AB→C
C→A
BC→D
CD→B
D→E
D→G
BE→C
CG→D
CE→G

Couverture minimal

Il peut y avoir plusieurs couvertures minimales:
si on élimine CE→A, puis CG→D, puis ACD→B
on obtient:

AD→C
D→G
C→A BE→C
BC→D

Autre couverture

Décomposition par jointure conservant l'information

Soit la relation R avec les attributs A_1 ……A_k.
On considère:

- Des ensembles d'attributs $X_1,X_2,\ldots\ldots, X_n$ tels que $X_1 \cup X_2 \cup \ldots\ldots \cup X_n = \{ A_1,\ldots\ldots,A_k \}$
- Les projections de R sur $X_1, \ldots\ldots, X_n$. $proj_{X1}(R),\ldots\ldots, proj_{Xn}(R)$

On veut remplacer R par ses n projections.
Une condition importante est de pouvoir retrouver R à partir de ses projections.
La seule façon de faire est de considérer la jointure :

$$m_{X_1,\ldots\ldots,X_n}(R) = proj_{X_1}(R) \bowtie proj_{X_2}(R) \bowtie \ldots\ldots \bowtie proj_{X_n}(R)$$

La condition importante est donc que:
$R = m_{X_1,\ldots\ldots,X_n}(R)$ on dit alors que l'on a décomposé R sans perte d'information.

Remarque:
On a toujours:

- $R \subseteq m_{X_1,\ldots\ldots,X_n}(R)$
- $m_{X_1,\ldots\ldots,X_n}(m_{X_1,\ldots\ldots,X_n}(R)) = m_{X_1,\ldots\ldots,X_n}(R)$
- $proj_{X_i}(m_{X_1,\ldots\ldots,X_n}(R)) = proj_{X_i}(R)$

On verra plus tard un test efficace pour savoir si $R = m_{X_1,\ldots\ldots,X_n}(R)$ compte tenu des seules dépendances connues.

Problème:
Une décomposition peut préserver l'information mais pas les dépendances:
Soit F un ensemble de dépendances explicitées pour R on en déduit:
$\pi_{X_i}(F)$ = les $X \rightarrow Y \in F^+$ telles que $XY \subseteq X_i$.

On considère:
$(\pi_{X_1}(F) \cup \pi_{X_2}(F) \cup \ldots\ldots \cup \pi_{X_n}(F))^+ = ?F^+$
Si oui, on dit que la décomposition préserve les dépendances

Exemple:

Soient les relations:
V Ville
R Rue
C Code postal
On a $F=\{C \rightarrow V,\ VR \rightarrow C\}$

Soit une table T d'attributs V,R,C
On veux décomposer VRC en RC et VC, on a bien:
$proj_{RC}(T) \bowtie proj_{VC}(T) = T$

On a :
$\pi_{RC}(F)$ = les dépendances triviales $RC \rightarrow R,\ RC \rightarrow R,\ R \rightarrow R,\ C \rightarrow C$
$\pi_{VC}(F)$ = les dépendances triviales et $C \rightarrow V$
Donc $(\pi_{RC}(F) \cup \pi_{VC}(F))^+$ ne contient pas $VR \rightarrow C$

Cours de Base de données

17.04.2000

Chapitre 4. Dépendances fonctionnelles et formes normales (suite)

Soit un schéma relationnel avec les attributs A_1 ……A_k.

- La décomposition d'une relation par jointure qui conserve l'information est de la forme :
  $E_1 \cup E_2 \cup \ldots\ldots \cup E_p = \{A_1, \ldots, A_n\}$

La jointure $proj_{E1}(R) \bowtie \ldots\ldots \bowtie proj_{En}(R) \supseteq R$.
Et s'il y a égalité chaque fois que R vérifie les dépendances fonctionnelles de F, on dit que la décomposition conserve l'information relativement à F.

- Test pour savoir si une décomposition conserve l'info.

1. Construire une table de k colonnes (des attributs) et p lignes (les projections)
2. Mettre dans la colonne j de la ligne i la valeur a_j si l'attribut j est dans E_j, sinon mettre b_{ij}
3. Répéter jusqu'à stabilité de la table :
   - Choisir une dépendance $X \rightarrow Y$ de F telles que 2 lignes i_1 et i_2 coïncident sur Y et égales leurs valeurs sur Y

La décomposition est sans perte d'information si à la fin il y a une ligne avec a_1 ……a_k

- Décomposition d'une relation par jointure qui préserve les dépendances fonctionnelles de F

  $\Pi_{E1}(F)$ = les $X \rightarrow F$ de F^+ ou $X \cup Y \in E_1$
  $\Pi_{Ep}(F)$ = les $X \rightarrow F$ de F^+ ou $X \cup Y \in E_p$

On a toujours :

$\Pi_{E1}(F) \cup \Pi_{E2}(F) \cup \ldots\ldots \cup \Pi_{Ep}(F) \subseteq F^+$

Mais s'il y a égalité alors la décomposition conserve les dépendances.

Exemple :

Soient les relations :
V Ville, R Rue, C Code postal.
Et les dépendances fonctionnelles : F={C→V, VR→C}.

Quelle décomposition E1=RC, E2=VC conserve l'info ?

	V	**R**	**C**
E1=RC	b1,1 → a1	a2	a3
E2=VC	a1	b2,2	a3

On regarde les dépendances fonctionnelles.

Les deux lignes sont égales sous C, on égale donc les éléments de la colonne V ; on obtient une ligne avec que des a.
D'après le test, cela veut dire que l'on conserve l'information.

! !Mais on ne préserve pas les dépendances.

En effet : $\Pi_{RC}(F) = \varnothing^+$ (= les trivialités du style : $X \rightarrow X$ ou encore $Y \subseteq X \subseteq \{R,C\}$)

$\Pi_{VC}(F) = \{C \rightarrow V\}^+$

mais $\Pi_{RC}(F) \cup \Pi_{VC}(F) = \{C \rightarrow V\}^+ \not\subset VR \rightarrow C$

Exemple d'état des tables :

V	R	C
Paris	10 rue Machin	75010
Paris	10 rue Machin	75011

Ne vérifie pas VR → C

R	C
10 rue Machin	75010
10 rue Mochin	75011

Satisfait $\Pi_{RC}(F)$.

V	C
Paris	75010
Paris	75011

Satisfait $\Pi_{VC}(F)$.

Problématique des formes normales.

Soient les donnée suivantes :
Relation, Fournisseur(Nom, Adresse, Article, Prix).
F = {Nom, Ar → Pr, Nom → Ad }

Cette relation présente des redondances ; l'adresse du fournisseur est répétée pour chaque article qu'il livre.
Gaspillage de mémoire, augmentation du temps d'accès !
De plus risque d'anomalies lors de mise à jour ; le fournisseur change d'adresse, innovation dans le catalogue, suppression d'un article ou d'un fournisseur.

Solution : Eclater la relation en deux.
(Nom, Adresse) et (Nom, Article, Prix).

Mais que deviennent les dépendances et l'info ?

(R1) $\Pi_{Nom, Ad}(F) = \{Nom \rightarrow Adresse\}^+$
(R2) $\Pi_{Nom, Art,Prix}(F) = \{Nom, Art \rightarrow Prix\}^+$

} F

	Nom	Adresse	Article	Prix
R1	a1	a2	b1,3	b1,4
R2	a1	B2,2 → a2	a3	a4

Pour tester si les dépendances sont conservées :

1ere façon : Calculer $(\Pi_{E1}(F) \cup \ldots \cup \Pi_{Ep}(F))^+$ Mauvais car s'exécute en temps exponentiel.

2eme façon : Pour chaque X →Y de F calculer X^+ au sens de $\Pi_{E1}(F) \cup \ldots \cup \Pi_{Ep}(F)$ et tester si $Y \subseteq X^+$

Chapitre 5. Forme normale de Boyce-Codd.

Définition :
Le schéma relationnel est en forme normale de B-C si :
Pour toute dépendance fonctionnelle $X \rightarrow A$ de F^+ ou $A \notin X$, les prémisses forment une super clé de la relation.
(c'est à dire que $X \rightarrow$ tout attribut est dans F^+)

Dans ce cas, il n'y plus de redondances.
En effet supposons $X \rightarrow A$ une redondance ;

X	Y	A
x	y_1	a
x	y_2	a

En cas de forme normale de B-C on aurait $(X \rightarrow Y) \in F^+$ et donc $y_1 = y_2$.

Algorithme de décomposition en forme normale de B-C conservant l'information :
Hélas on ne pourra pas conserver les dépendances.

Lemme1 :

Si la décomposition $(E_1, \ldots, E_p)$ de $\{A_1, \ldots, A_n\}$ conserve l'information (bien sur relativement à un ensemble de dépendances), et si la décomposition (S_1, S_2) de E_1 conserve l'information relative à $\Pi_{E1}(F)$ alors la décomposition $(S_1, S_2, E_2, \ldots, E_p)$ de $\{A_1, \ldots, A_n\}$ conserve l'information relative à F.

Preuve : Associativité des produits de jointures.

$proj_{E1}(R) \bowtie \ldots \bowtie proj_{En}(R) = R$.
De plus :

$proj_{S1}(proj_{E1}(R)) = proj_{S1}(R)$ et
$proj_{S2}(proj_{E1}(R)) = proj_{S2}(R)$

D'où :

$proj_{S1}(proj_{E1}(R)) \bowtie proj_{S2}(proj_{E1}(R)) = proj_{E1}(R)$

Lemme2 :

1. Toute relation à deux attributs est en FNBC.
2. Si $R=\{A_1, \ldots, A_n\}$ n'est pas en FNBC, alors il existe des attributs A,B tels que : $(R \setminus AB \rightarrow A) \in F^+$.

Preuve :

1. $R=A_1, A_2$ si $(X \rightarrow A_1) \in F^+$ et $A_1 \notin X$ alors $X=A_2$
   donc $(A_2 \rightarrow A_1) \in F^+ \Rightarrow A_2$ est une clé.
2. Si R non FNBC, il existe $(X \rightarrow a) \in F^+$, $A \notin X$, X ne contient pas de clé, donc $X \neq R \setminus A$.
   Soit $B \notin X, A \Rightarrow R \setminus AX \supseteq X$ et $\Rightarrow R \setminus AB \rightarrow A$

Remarque :
La réciproque du 2. est fausse.
par exemple soient : $R=ABC$, $F=\{C \rightarrow A, C \rightarrow B\}$, C une clé
R est en FNBC mais $RAB=C \rightarrow A$.

Algo1 :

entrée : R, dépendances fonctionnelles F telles que $\exists A,B\ (R \setminus AB \rightarrow A) \in F^+$.

sortie : décomposition conservant l'info de la forme : R \ A, XA avec XA en FNBC.

Corps de l'algo : (pseudo PASCAL)

init : Z :=R
loop : tant que ($\exists$ une paire A,B | $(Z \setminus AB \rightarrow A) \rightarrow F^+$)
faire Z :=Z\B
fin : A= le dernier A obtenu dans la boucle
X= le dernier Z auquel on enlève A.

À la fin, on a :
$Z \setminus A \rightarrow A$ avec Z :=Z \ B c'est à dire (nouveau Z) \ $A \rightarrow A$.
On pose (nouveau Z) = Z \ B = X : donc,
$X \rightarrow A$, Z = XA est en FNBC car le lemme2 ne s'applique plus.
$(R \setminus A) \cap (XA) = X \rightarrow A = (((R \setminus A) \setminus XA) \cup (XA \setminus (R \setminus A))$

Algo. de décomposition avec conservation de l'info en relations toutes en FNBC :

init : Z :=R
loop : tant que (l'algo1 appliqué à Z)
faire (appliquer l'algo1 pour sortir Z\A et XA, ajouter XA à la liste, poser Z :=Z\A)
fin : rajouter Z à la liste.
Cette liste donne la décomposition recherchée.

Les XA rajoutés à la liste lors de la boucle sont tous en FNBC.
Les Z rajoutés à la liste lors de la boucle sont tous en FNBC.
Car la boucle ne pouvant continuer c'est que l'hypothèse du lemme2 est fausse et donc que Z est en FNBC.

Le fait que cette décomposition conserve l'info. vient de ce que l'algo1 conserve l'info., et que le lemme1 permet de voir qu'a chaque étape la décomposition de liste + Z conserve l'info.

Chapitre 6. La 3éme Forme normale.

Définition :
R est en 3eme Forme Normale si $\forall(X \rightarrow A)$ de F^+ avec $A \notin X$:
- ou bien X est une super-clé,
- ou bien A est un élément d'une clé de R.

Algo. de décomposition :

Considérer une couverture minimale de F.
1^{er} cas : L'une des dépendances est de la forme $R \setminus A \rightarrow A$ alors R ok.
2^{eme} cas : Sinon, la décomposition est formée des XA ou $X \rightarrow A$ est dans la couverture minimale et d'une clé de R.

Proposition :
La décomposition ainsi obtenue préserve l'information ET conserve les dépendances.

Cours de Base de données

15.05.2000

Chapitre 4. Dépendances fonctionnelles et formes normales (fin)

Soit $X \rightarrow Y$, avec X, Y des ensembles d'attributs, la relation R vérifie la dépendance fonctionnelle $X \rightarrow Y$ si 2 lignes quelconques qui coïncident sur les colonnes de X coïncident aussi sur celles deY.

En pratique on postule un certain nombre de dépendances fonctionnelles, d'où une famille F de dépendances fonctionnelles.
Il s'agit de contraintes qu'on impose à tous les états de la BD.
De F on déduit F^+=toutes les dépendances fonctionnelles que l'on peut "*déduire*" (2 sens) de F.

1er sens de *déduire*:
une dépendance fonctionnelle $Z \rightarrow T$ se déduit de F si toute relation qui satisfait toutes les dépendances de F satisfait aussi $Z \rightarrow T$.

2eme sens de *déduire*:
Une dépendance fonctionnelle $Z \rightarrow T$ se déduit de F s'il existe une preuve de $Z \rightarrow T$ à partir des dépendances fonctionnelles de F utilisant les axiomes et règles d'Armstrong.

Rappel:
Axiome de réflexivité: Si $Y \subseteq X$ alors $X \rightarrow Y$
Règle d'accroissement : De $X \rightarrow Y$ on déduit $XZ \rightarrow YZ$ pour tout Z
Règle de transitivité: De $X \rightarrow Y$ et $Y \rightarrow Z$ on déduit $X \rightarrow Z$

Théorème
Ces 2 sens sont en fait identique

$X^+_{\text{(par rapport à F)}} = \{A: X \rightarrow A \text{ est dans } F^+\}$ en pratique, c'est toujours les X^+ qu'on calcule

- Recouvrement minimal des dépendances:

F un ensemble de dépendances fonctionnelles est minimal si:
1. Toute dépendance fonctionnelle de F est de la forme $X \rightarrow A$, avec A attribut et X ensemble d'attributs.
2. Si $X \rightarrow A$ est dans f alors $X \rightarrow A \notin (F \setminus \{X \rightarrow A\})^+$ en d'autre termes $(F \setminus \{X \rightarrow A\})^+ \not\supset F^+$
3. Si $X \rightarrow A$ est dans F alors $X \rightarrow A \in [(F \setminus \{X \rightarrow A\}) \cup \{X' \rightarrow A\}]^+$ si $X' \not\supset X$
   En d'autre terme $((F \setminus \{X \rightarrow A\}) \cup \{X' \rightarrow A\})^+ \not\supset F^+$

} Ce n'est pas unique en général mais pour tout F il existe G tel que G minimal et $G^+ = F^+$

- Décomposition d'un schéma relationnel

Données: ensemble d'attributs $\{A_1, \ldots\ldots, A_n\}$, famille F
On cherche une famille $E_1, \ldots\ldots, E_p$ de parties de $\{A_1, \ldots\ldots, A_n\}$ telle que : si R est une relation d'attributs $A_1, \ldots\ldots, A_n$ qui vérifie les dépendances fonctionnelles de F alors
$R = proj_{E1}(R) \bowtie proj_{E2}(R) \bowtie \ldots\ldots proj_{Ep}(R)$

On dit alors que la décomposition est sans perte d'information et, si possible
$(\Pi_{E1}(F) \cup \Pi_{E2}(F) \cup \ldots\ldots \cup \Pi_{Ep}(F))^+ = F^+$
$\Pi_E(F)$= les $X \to Y$ de F^+ tels que $XY \subseteq E$ = les dépendances fonctionnelles de F^+ portant sur les seuls attributs dans E

Algo pour tester si une décomposition préserve l'information relativement à F

1. On construit une table de colonnes indexées par les attributs
   avec p ligne
   Sur la ligne i on écrit a_j en colonne j si l'attribut A_j est dans E_i
   b_{ij} en colonne j si l'attribut $A_j \notin E_i$
2. Répéter sur la table les opérations suivantes :
   Pour chaque dépendances fonctionnelles $X \to Y$ de F choisir 2 lignes qui coïncident sur les colonnes de X et égaler les coefficient de ces lignes sur les colonnes de Y.
3. La décomposition est sans perte d'information i et seulement si la table finale contient la ligne $(a_1, \ldots\ldots a_n)$

Algo pour tester si une décomposition préserve les dépendances
Ne pas chercher à calculer $(\Pi_{E1}(F) \cup \Pi_{E2}(F) \cup \ldots\ldots \cup \Pi_{Ep}(F))^+ = F^+$
Pour chaque $X \to Y$ de F calculer le X^+ relativement à $\Pi_{E1}(F) \cup \Pi_{E2}(F) \cup \ldots\ldots \cup \Pi_{Ep}(F)$
et tester si $X^+ \supseteq Y$

Cas particulier où p=2
Fait (E1, E2) est sans perte d'information (relativement à f) si et seulement si :
$E_1 \cap E_2 \to E_1 \Delta E_2$ est dans F^+

$E_1 \Delta E_2$ = les attributs qui sont dans exactement dans un seul d'entre E1 et E2
$= (E_1 \backslash E_2) \cup (E_2 \backslash E_1)$
Δ = différence symétrique

Chapitre 5. Forme normale de Boyce-Codd.

X	Y	A
x	y_1	a
x	y_2	?

attributs

C'est nécessairement

$X \to A$ donc X est une superclé si $X \to A$ donc $X \to Y$ donc les 2lignes sont égales

On peut avoir intérêt à casser la relation en 2 relations, l'une avec les attributs X et Y l'autre avec les attributs X etA

Définition:
$\{A_1, \ldots\ldots, A_n\}$, F ensemble de dépendance fonctionnelle. Ce schéma est en forma normale de Boyce-Codd (BCNF) si toute dépendance $X \to A$ de F+ ou $A \notin X$ alors X est une superclé
$(X \to \{A_1, \ldots\ldots, A_n\} \in F^+)$

Lemme 1:
Si E_1, ……, E_p est sans perte d'information relativement à F
Si S_1, S_2 est sans perte d'information relativement à $\Pi_{E1}(F)$
Alors S1, S2, E_1, ……, E_p est sans perte d'information relativement à F

Lemme 2:
1. **Tout schema à 2 attributs est en FNBC**
2. **Si ($\{A_1, ……, A_n\}$, F) n'est pas en FNBC alors il existe A, B tels que $\{A_1, ……, A_n\}\backslash\{A, B\}\rightarrow A$**

Algo de décomposition d'un schémas ($\{A_1, ……, A_n\}$, F) en schéma satisfaisant la FNBC
Ceci sans perte d'information (mais les dépendances fonctionnelles ne sont pas toujours préservables)

Algo 1:
Décompose sans perte d'information d'un schéma ($\{A_1, ……, A_n\}$, F) non en FNBC en 2 schémas $\{A_1, ……, A_n\}\backslash\{A\}$ et XA tels que $X\rightarrow A\in F^+$ et XA est en FNBC (relativement à $\Pi_{XA}(F)$)

Initialisation	$Z=\Pi_{E1}(F)$
Boucle	Tant qu'il existe une paire A, B de Z telle que $Z\backslash AB\rightarrow A\in F^+$ poser $Z=Z\backslash B$
Terminaison	Poser A= le dernier A obtenu, X = (le dernier Z obtenu)\A

Preuve de correction:
1) Ca se termine en au plus n étape car Z décroît
2) A la fin on a $X\rightarrow A$ par construction et Z=XA est en FNBC à cause du lemme 2
3) Pas de perte d'information car
   $\{A_1, ……, A_n\}\backslash\{A\} \cap XA = X$ et
   $(\{A_1, ……, A_n\}\backslash A) \Delta XA = A$ et $X\rightarrow A\in F^+$

Algo 2:
Décomposition sans perte d'information de ($\{A_1, ……, A_n\}$, F) en une suite de schémas en FNBC

Initialisation	$Z=\{A_1, ……, A_n\}$, L = liste vide
Boucle	tant que Z n'est pas en FNBC appliquer l'algo 1 Rajouter XA à L et poser $Z=Z\backslash A$
Fin	Rajouter Z à L, la liste L est la décomposition cherchée

Exemple:

{Cours, Professeur, Heure, Salle, Diplôme, Etudiant}
F={C→P, HS→C, HP→S, CE→D, HE→S}
Algo 2

Z=CPHSDE	AB=CP	HSDE→C (car HS→C dans F)
Z=CHSDE	AB=SC	HDE→S (car HE→S est dans F
Z=HSDE	terminé→ décomposition sans perte (HSE, CPHDE) en FNBC	

Z=CPHDE	AB=PH	CDE→P (car C→P est dans F)
Z=CPDE	AB=PE	CD→D (car C→P est dans F)
Z=CPD	AB=PD	C→P (car C→P est dans F)
Z=CP	terminé→décomposition sans perte de CPHDE en (CP, CHDE) en FNBC	

Z=CHDE	AB=DH	CE→D
Z=CDE	terminé→décomposition sans perte de CHDE en 5CDE, CHE)	

Z=CHE	terminé d'ou

(HSE, CP, CDE, CHE) = décomposition en FNBC

➢ 3eme Forme Normale

({A_1, ……, A_n}, F) est en 3eme Forme Normale si pour toute dépendance fonctionnelle X→A, de F+ ou bien X est une superclé ou bien A est élément d'au moins une clé

Algo de décomposition en 3eme Forme Normale

1) déterminer une couverture minimale G
2) déterminer une clé
3) considéré la décomposition forme des XA où (X→A)∈G et de la clé

Cette décomposition est sans perte d'information et préserve les dépendances