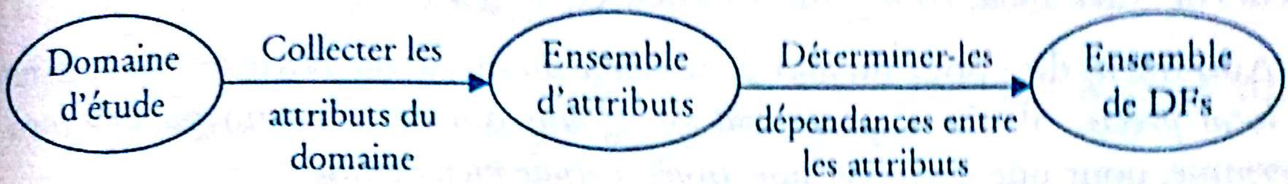
2. Les dépendances fonctionnelles (DF)

2.1. Notion de DF :



Soit le sous-ensemble d'attributs tirés du dictionnaire de données relatif à la gestion d'une scolarité :

Nom d'attribut	*Signification*
Cod_mod	Code du module
Num_ens	Numéro d'enseignant
Jour	Jour de semaine
Heure	Horaire d'une séance
Section	Numéro de la section
Local	Numéro de la salle de cours
An_étude	Année d'étude

Sur la base de cet ensemble d'attributs, nous construisons une relation qui nous renseigne sur l'emploi du temps, d'une filière donnée (informatique par exemple, toutes années confondues) dont nous donnons une partie en extension (quelques exemples de tuples).

Emploi du temps :

Jour	*Heure*	*Local*	*Cod_mod*	*Num_ens*	*An_étude*	*Section*
Samedi	$8^{h}00$	L1	Algo	E1	2	A
Samedi	$13^{h}00$	L2	Archi01	E2	2	A
Lundi	$9^{h}30$	L1	Algo	E4	2	B
Mardi	$11^{h}20$	L3	Log	E3	2	B
Samedi	$13^{h}00$	L3	Sys01	E4	3	A
...						

2.2. Propriétés des DF :

Dans cette partie, X et Y représentent deux sous-ensembles d'attrib[uts] d'une relation R.

- **Réflexivité :**

Pour tout $Y \subseteq X$, on a $X \rightarrow Y$, en particulier $X \rightarrow X$

- **Augmentation :**

Si $X \rightarrow Y$ alors $X \cup Z \rightarrow Y \cup Z$

Exemple :

Cod_mod → Libellé_mod ⇒
Cod_mod, Coef → Libellé_mod, Coef

- **Transitivité :**

Si $X \rightarrow Y$ et $Y \rightarrow Z$ alors $X \rightarrow Z$

Exemple :

Num_ens → grade et grade → salaire de base
⇒ Num_ens → salaire de base

Les trois règles présentées ci-dessus sont connues sous le nom d'axiom[es] d'ARMSTONG, d'autres règles peuvent en être déduites : il s'agit de l'union, la pseudo-transitivité et la décomposition

- **Union :**

Si $X \rightarrow Y$ et $X \rightarrow Z$ alors $X \rightarrow Y \cup Z$

Exemple :

Jour, Heure, Local → Cod_mod et Jour, Heure, Local → Num_ens
⇒ Jour, Heure, Local → Cod_mod, Num_ens

Après cette présentation informelle de la notion de dépendance fonctionnelle, nous donnons la définition formelle suivante.

➡ Définition (Dépendance Fonctionnelle) :

Soit R (a_1, a_2, ..., a_n) ($n \geq 2$) un schéma de relation, soit X, Y, Z trois sous- ensembles d'attributs de A $= \{a_1, a_2, \ldots\ldots, a_n\}$ tels que :

$X \cap Y \cap Z = \emptyset$ et $X \cup Y \cup Z = A$ (Z peut être vide)

On dira que *X détermine Y*, ou *Y dépend fonctionnellement de X* et on notera $X \rightarrow Y$ si : quels que soient les tuples (x, y, z) et (x', y', z') de R, $x = x' \Rightarrow y = y'$

Autrement dit : *une valeur de x détermine une et une seule valeur de y*, ou bien connaissant la valeur de x, nous pouvons déterminer la valeur de y sans ambiguïté.

Exemples :

- Dans une relation qui décrit les modules enseignés *R (<u>Cod_mod</u>, Libellé_mod, Coef)* où Libellé_mod et Coef désignent respectivement le nom d'un module et son coefficient, nous avons les DFs suivantes :

  Cod_mod → Libellé_mod

  Cod_mod → Coef

  Connaissant *Cod_mod*, nous pouvons déterminer son *Libellé* et son *Coefficient.*
- Dans une relation qui décrit les enseignants, nous avons les DFs suivantes :

  Num_ens → Grade

  Num_ens → Nom_ens

  Num_ens → Prénom_ens

  On pourra les écrire aussi comme suit :

  Num_ens → Grade, Nom_ens, Prénom_ens

  Nom_ens, Prénom_ens et Grade désignent respectivement le nom, le prénom et le grade d'un enseignant.

  Dans une relation qui décrit les sections, nous avons :

  Section, An_étude → Nbre_étudiants

De là, nous constatons ce qui suit :

Connaissant le *jour de semaine*, *l'heure* et le *local* (i.e les valeurs de attributs *jour, heure, local*), nous pouvons déterminer de façon unique les valeurs de *Cod_mod, Num_ens, An_étude* et *Section*)).

Autrement dit : pour un *jour* de semaine *donné*, à une *heure précise* et dans un *local précis*, il n'y a qu'*un seul enseignant* qui enseigne un *module bien déterminé*, pour une *section* et une *année d'étude bien définie*.

On dira que {*Jour, Heure, Local*} **déterminent** *Cod_mod, Num_ens, An_étude* et *Section*.

On dira aussi que *Cod_mod, Num_ens, An_étude* et *Section* ***dépendent fonctionnellement*** de {*Jour, Heure, Local*}.

On écrira : *Jour, heure, local* → *Cod_mod, Num_ens, An_étude, Section*

L'ensemble des attributs de la *partie gauche détermine* les attributs de la *partie droite*.

Ou encore

Jour, heure, local → *Cod_mod*

Jour, heure, local → *Num_ens*

Jour, heure, local → *An_étude*

Jour, heure, local → *Section*

Nous constatons aussi que *Cod_mod* ***ne détermine pas*** *Num_ens*. En effet, pour le module **Algo**, nous avons deux enseignants différents E1 et E4

Connaissant le module, nous ne pouvons pas déterminer l'enseignant.

De même, *Num_ens* ne détermine pas *Cod_mod*, du fait que l'enseignant E4 assure deux modules différents *Algo* et *sys01*.

Remarques

- *Une DF est une assertion qui est définie sur toutes les réalisations (tuples) d'une relation et pas sur un tuple particulier.*
- *Une DF traduit une certaine perception de la réalité, elle correspond à une contrainte sur les données, qui doit être vérifiée en permanence*

- **Pseudo-transitivité :**

Si $X \rightarrow Y$ et $Y \cup W \rightarrow Z \Rightarrow X \cup W \rightarrow Z$

Exemple :

> *Num_ens → Grade*
> *Grade, Nom_ens → Salaire*
> ⇒ *Num_ens, Nom_ens → Salaire*

- **Décomposition :**

Si $X \rightarrow Y$ et $Z \subseteq Y$ alors $X \rightarrow Z$

Exemple :

> *Cod_mod → Libellé_mod, Coef*   *{Coef} ⊆ {Libellé_mod, Coef}*
> ⇒ *Cod_mod → Coef*

2.3. Typologie des DF :

- **DF triviale :**

$X \rightarrow Y$ est triviale si $Y \subseteq X$. (Réflexivité)

- **DF élémentaire :**

Une DF $X \rightarrow Y$ est dite élémentaire si $\forall X' \subseteq X, X' \nrightarrow Y$.

Autrement dit : Y ne dépend pas d'une partie de X.

Exemple :

> *Jour, Heure, Local → Cod_mod est élémentaire.*
> *Cod_mod, Libellé_mod → Coef* n'est pas élémentaire car *Cod_mod* suffit pour déterminer *Coef*.

Cette notion de DF élémentaire sert à conserver les seules DF utiles.

- **DF canonique :**

Une DF X → Y est dite canonique si son membre droit est réduit à un singleton (i.e un seul attribut).

Notons que toute DF qui n'est pas canonique peut être transformée en un ensemble de DFs canoniques par décomposition

Exemple :

La DF *Num_ens → Grade, Nom_ens, Prénom_ens* n'est pas canonique, elle donne lieu à trois DFs canoniques.

Num_ens → Grade

Num_ens → Nom_ens

Num_ens → Prénom_ens

- **DF directe :**

X → Y est directe si :

- Elle est élémentaire.
- Y ne dépend pas transitivement de X

(i.e $\nexists$ Z/ X → Z et Z → Y)

2.4. Couverture minimale (irréductible) :

- **Définition :**

La couverture minimale d'un ensemble de DF est un sous-ensemble minimum de DF élémentaires permettant de générer toutes les autres.

- **Théorème :**

Tout ensemble de DF admet une couverture minimale, en général non unique.

Exemple :

1. *Cod_mod → Cod_filière*
2. *Cod_filière → Libellé_filière*
3. *Cod_mod → Libellé_filière*
4. *Jour, Heure, Local → Num_ens*
5. *Jour, Heur, Local → Cod_filière*
6. *Jour, Heure, Local → Section*
7. *Jour, Heur, Local → Groupe*
8. *Jour, Heure, Local → An_étude*
9. *Jour, Heure, Local → Cod_mod*
10. *Jour, Heure, Local, Num_ens → Cod_filière*

La couverture minimale de cet ensemble de DF consiste à éliminer les DF numéro 3 (obtenu par transitivité de 1 et 2) numéro 10 obtenue par augmentation de la DF numéro 5, et la DF numéro 5 obtenue par transitivité de 1 et 9.

Matricule, Nom_et est une clé de la relation ***Etudiant***, mais elle n'est pas minimale car il suffit d'avoir le matricule de l'étudiant, pour déterminer tous les autres attributs, y compris le nom.

Matricule → Nom_et

Matricule → Prénom_et

Matricule → Dat_nais_et

Matricule → Adresse_et

Matricule est alors *une clé minimale* de la relation ***Etudiant***. Le schéma de relation est :

Etudiant (Matricule, Nom_et, Prénom_et, Dat_nais_et, Adresse_et)

Dans une relation décrivant les résultats obtenus par les étudiants dans les différents modules, nous aurons :

Résultats (Matricule, Cod_mod, Moyenne)

{Matricule, cod_mod} est une clé de la relation ***Résultats***. De plus, elle est minimale.

➡ **Clés candidates/Clé primaire :**

Une relation peut avoir plusieurs clés possibles, on parle alors de *clés candidates*, on en choisit une qu'on appelle ***clé primaire.***

Dans l'exemple de la relation ***Emploi du temps***, on peut vérifier que *{Jour, Heure, Local}* et *{Jour, Heure, Num_ens}* sont deux ***clés candidates*** de la relation.

Si on choisit *{Jour, Heure, Local}*, elle devient ***clé primaire*** et le schéma de la relation s'écrit :

Emploi du temps (Jour, Heure, Local, Num_ens, Cod_mod, Section, An_étude)

On peut aussi choisir *{Jour, Heure, Num_ens}*, elle devient à son tour clé ***primaire*** et le schéma de relation devient :

Emploi du temps (Jour, Heure, Num_ens, Local, Cod_mod, Section, An_étude)

2.6. Notion de clé de relation :

De façon informelle, la clé d'une relation R est un ensemble d'attributs (un ou plusieurs) de R, qui détermine tous les attributs de R.

- **Définition (clé de relation) :**

Soit A l'ensemble des attributs d'une relation R et soit X un sous-ensemble de A. X est une clé de R si $\forall$ a $\in$ A, X $\rightarrow$ a.

Reprenons l'exemple de la relation Emploi du temps

Jour, Heure, Local $\rightarrow$ *Num_ens*

Jour, Heure, Local $\rightarrow$ *Cod_mod*

Jour, Heure, Local $\rightarrow$ *Section*

Jour, Heure, Local $\rightarrow$ *An_étude*

Donc {*Jour, Heure, Local*} est une ***clé de la relation Emploi du temps***, elle sera écrite juste après le nom de la relation et soulignée pour la distinguer des autres attributs qui ne sont pas clés. On aura la relation :

Emploi du temps (<u>Jour, Heure, Local</u>, Num_ens, Cod_mod, Section, An_étude)

De même dans la relation Module, *Cod_mod* est une clé. On écrira

Module (<u>Cod_mod</u>, Libellé_mod, Coef)

- **Clé minimale :**

Soit X un sous-ensemble d'attributs d'une relation R. X est dite ***clé minimale***, si X est une clé de R et si toute DF X $\rightarrow$ a de R *est élémentaire.*

Exemple :

Dans une relation décrivant les étudiants :

Etudiant (Matricule, Nom_et, Prénom_et, Dat_nais_et, Adresse_et)

Matricule, Nom_et $\rightarrow$ *Prénom_et*

Matricule, Nom_et $\rightarrow$ *Dat_nais_et*

Matricule, Nom_et $\rightarrow$ *Adresse_et*

Matricule $\rightarrow$ *Nom_et*

5. Fermeture transitive :

- **Définition :**

On appelle fermeture transitive d'un ensemble F de DF, l'ensemble constitué de F lui-même et de l'ensemble des DF déduites par transitivité.

- *Exemple :*

Soit l'ensemble de DF, F = {*Num_ens → Grade,*
Grade → Salaire
Grade → Nbre_Heures}

La fermeture transitive de F sera :

F+ = F ∪{*Num_ens, → Salaire, Num_ens → Nbre_Heures}*
= {*Num_ens → Grade*
Grade → Salaire
Grade → Nbre_ Heures
Num_ens → Salaire
Num_ens → Nbre_Heures}

Dans ce cas la couverture minimale de F+ est F

Remarque

La recherche de la couverture minimale d'un ensemble de DF est un élément essentiel dans le processus de normalisation, afin de décomposer une relation en plus petites relations.

- Superclé :

C'est tout ensemble d'attributs contenant tous les attributs de la clé primaire de la relation.

En particulier, l'ensemble de tous les attributs d'une relation constitue une superclé. La clé primaire d'une relation est une superclé aussi.

- Clé étrangère (extérieure) :

La notion de clé étrangère sert à faire le lien entre deux relations d'une même BD, et permet au SGBDR de maintenir la cohérence des lignes (tuples) de deux relations ou d'une même relation.

- Définition :

Soit A un ensemble d'attributs qui constitue la clé primaire d'une relation R1. A est dite clé étrangère de R2, si :

1. A apparaît dans R2.
2. Une valeur de clé extérieure d'un tuple quelconque de R2 est soit "*NULL*", soit apparaît comme valeur de clé primaire d'un tuple de R1.

Exemple :

Module (*Cod_mod, Libellé_mod, Coef, An_étude, Cod_filière*)

Filière (*Cod_filière, Libellé_filière*)

Cod_filière est une clé primaire dans la relation Filière et elle est étrangère dans la relation module.

3. Les formes normales

La théorie de normalisation est destinée à concevoir correctement l schéma d'une BD relationnelle, c'est à dire :

- sans redondance d'informations
- sans anomalies de mise à jour

Expliquons ceci sur un exemple :

Considérons la relation *Etud_mod* suivante :

Etud_mod (<u>Matricule, Cod_mod</u>, Nom_et, Prénom_et, Moyenne) po laquelle nous présentons une partie en extension. (l'attribut *Moyenne* désig la moyenne d'un étudiant dans un module).

Matricule	*Nom_et*	*Prénom_et*	*Cod_mod*	*Moyenne*
52000001	ABDI	Mohamed	Algo	10
52000001	ABDI	Mohamed	Archi 01	12.5
52000001	ABDI	Mohamed	Logique	11
52000002	BADAOUI	Karim	Algo	8.5
52000002	BADAOUI	Karim	Archi 01	15
52000002	BADAOUI	Karim	Logique	11.75
⋮				

Nous remarquons tout de suite une ***redondance*** dans ce tableau, en effe chaque tuple pour l'étudiant 'ABDI' nous indique son matricule, son nom e son prénom. Idem pour les tuples de l'étudiant 'BADAOUI'. D'une maniè générale, le nom et le prénom du même étudiant sont répétés autant de fo que le nombre de modules auxquels il est inscrit.

Dans le modèle relationnel, on doit éviter ce type de situation (redondance d'informations), pour cela on effectue ce qu'on appelle une ***normalisation*** sur les relations, on obtient alors des relations normalisées On parle plus précisément de ***formes normales*** que nous présentons dans la suite.

3.1. Première forme normale : (1FN)

- **Définition (1FN) :**

Une relation R est en 1FN, si chacun de ses attributs a un *domaine atomique mono-valué*.

Autrement dit : pour chaque tuple de la relation, tout attribut possède une *seule valeur*.

Cette définition exclut les relations ayant des attributs dont les valeurs seraient des ensembles ou des listes de valeurs.

Exemple :

Reprenons l'exemple de la relation ***Résultats*** qui stocke les résultats des étudiants dans les modules. Supposons que l'on veuille sauvegarder pour chaque module et pour chaque étudiant, les différentes notes obtenues pendant l'année (pas uniquement la moyenne). Nous aurons à sauvegarder la note de l'EMD1, celle de l'EMD2, éventuellement celles de la synthèse et du rattrapage.

Soit le schéma de relation suivant :

Résultats (<u>Matricule, Cod_mod</u>, Nom_et, Prénom_et, Notes)

Pour l'étudiant « ABDI Mohamed », nous aurons par exemple 4 notes à sauvegarder, pour le module Algo :

52000001 ABDI Mohamed Algo {8 9.5 6.5 11}

Nous remarquons tout de suite que cette relation ***n'est pas en 1FN*** du fait que l'attribut ***notes*** possède quatre valeurs différentes, il n'est donc pas atomique.

Pour transformer cette relation en première forme normale (1FN) , il faut justement *décomposer cet attribut en quatre autres attributs* ***note1, note2, synt, ratt*** qui soient atomiques .

Nous aurons alors le schéma de relation suivant :

Résultats (<u>Matricule, Cod_mod</u>, Nom_et, Prénom_et, Note1, Note2, Synt, Ratt)

Chacun des attributs *Note1, Note2, Synt* et *Ratt* aura une valeur **atomique.** La relation ainsi obtenue *est en 1FN.*

Remarque

La première forme normale a pour objectif de mettre à plat les hiérarchies, dans un souci de simplicité

.2. Deuxième forme normale : (2FN)

Une relation R est dite en 2FN si et seulement si :

- Elle est en 1FN.
- Tout attribut n'appartenant pas à la clé ne dépend pas d'une partie de la clé. (i.e il est en dépendance fonctionnelle élémentaire de la clé).

Exemple :

La relation *Résultats (Matricule, Cod_mod, Nom_et, Prénom_et, Note1, Note2, Synt, Ratt)* n'est pas en 2FN

Car : *Matricule → Nom_et*
Matricule → Prénom_et

Nom_et et Prénom_et sont deux attributs qui dépendent d'une partie de la clé.

Il faut décomposer la relation *Résultats* en deux autres relations *Résultats (Matricule, Cod_mod , Note1, Note2, Synt, Ratt)* et *Etudiant (Matricule , Nom_et, Prénom_et)* ces deux dernières sont alors en 2FN.

Remarque

La 2FN n'est pas suffisante pour éviter toutes les redondances, nous pouvons voir ceci sur l'exemple suivant :
Considérons la relation Enseignant
Enseignant (Num_ens, Nom_ens, Prénom_ens, Dat_nais_ens, Adresse_ens, Grade, Nbr_heures)

Num_ens	Nom_ens	Prénom_ens	Dat_nais_ens	Adresse_ens	Grade	Nbr_heures
E1	SALAH	Mohamed	10/07/65	Alger	Chargé de cours	9
E2	EL_ALLAMI	Ryadh	25/03/59	Alger	Maître de conf.	7h $^{1/2}$
E3	NOUALI	Nora	15/08/662	Alger	Chargé de cours	9

Remarquons que le nombre d'heures d'enseignement dépend uniquement du grade de l'enseignant, en fait la même valeur sera répétée à chaque fois que l'on répète le même grade.

Nous avons un attribut (*Nbr_heures*) qui dépend d'un autre attribut non clé (Grade), ceci crée une autre forme de redondance. Pour l'éliminer, il faut mettre les relations en 3FN.

3.3. Troisième forme normale : (3FN)

Une relation est en 3FN si :

- Elle est en 2FN.
- Tout attribut n'appartenant pas à la clé ne dépend pas d'un attribut non clé.

Sur l'exemple précédent, il faudra supprimer l'attribut Nbr_heures de la relation Enseignant et créer une nouvelle relation Charge qui détermine le nombre d'heures d'enseignement, en fonction du grade de l'enseignant. On éclate la relation précédente en deux autres relations comme suit :

Enseignant (<u>Num_ens</u>, Nom_ens, Prénom_ens, Dat_nais_ens, Adresse_ens, Grade)

Charge (<u>Grade</u>, Nbr_heures)

Ainsi, ces deux relations constituent un schéma en *3FN*.

Remarque

> *La troisième Forme Normale a pour objectif d'éliminer les redondances dues aux DF transitives*

3.4. Forme normale de Boyce Codd (BCNF) :

Dans le cas où une relation admet plusieurs clés candidates, des anomalies peuvent subsister en 3FN. La forme normale de Boyce Codd permet d'éviter ces anomalies.

- **Définition :**

Une relation est en forme normale de *Boyce-Codd* (BCNF), si quelle que soit la dépendance fonctionnelle, le membre de gauche est une clé.

Remarque

> *Toute relation qui est en BCNF est forcément en 3FN.*

Exemple :

Soit la relation *R (Ville, Rue, Code)*, vérifiant l'ensemble de DF suivant :

Ville, Rue → Code

Code → Ville

Deux clés candidates peuvent être identifiées sur cette relation : *Ville, Rue* et *Code, Rue*

R est en 3FN, mais n'est pas en BCNF : le membre gauche de la DF *Code → Ville* n'est pas une clé.

Une extension possible de cette relation serait :

Ville	*Rue*	*Code*
Alger	Hassiba Ben Bouali	16014
Alger	Redha Houhou	16014

Remarquons qu'il y a une redondance entre *Code* et *Ville*.

Cependant, il n'est pas toujours possible de trouver une décomposition en relations BCNF, sans perte d'information, et qui préserve les DF.

La décomposition de R en

R1(Ville,Code), et *R2(Rue,Code)* évite la redondance, elle est sans perte d'information, mais elle ne préserve pas la DF : *Ville, Rue → Code*.