ENSICAEN
ÉCOLE NATIONALE SUPÉRIEURE D'INGÉNIEURS DE CAEN
& CENTRE DE RECHERCHE

1ère année Informatique

Bases de Données/PostgreSQL
- Analyse Merise -

- Cours -

Myriam Mokhtari-Brun

Conception des bases de données

1.Nécessité d'une méthode
2.La normalisation
3.La décomposition
4.La synthèse : modèle entités-associations

1. Conception de base de données relationnelle

- Lorsque l'on veut traduire un univers réel ou un schéma conceptuel par un modèle relationnel, on se pose la question suivante :
  - Quel est l'ensemble des relations qui vont être une représentation fidèle du schéma conceptuel (et donc du monde réel) sans que des problèmes de cohérence ne risquent de se poser ?

Nécessité d'une méthode

Avion			
AvNum	AvNom	AvCap	AvVille
0350	A320	200	Paris
0360	A320	200	Toulouse
0400	Caravelle	150	Nice
0420	B747	500	Lyon
0430	Mercure	100	Nantes
0450	Concorde	150	Paris
0500	A320	200	Paris

- Tous les avions de même nom ont la même capacité -> un lien entre nom d'avion et capacité.
- Du fait de ce lien, nous avons :
  - une redondance logique
  - des anomalies de stockage
  - des problèmes de reconnexion.

Redondance logique

- Un lien entre deux attributs d'une relation conduit à une duplication des valeurs du couple d'attributs.
- Dans l'exemple, on observe une redondance des valeurs du couple (AvNom, AvCap).
  - Qu'advient-il si on modifie la capacité de l'avion de numéro 0360 pour la porter à 220 places ?
    - L'ensemble contient (entre autres) les quadruplets (0350, A320, 200, Paris) et (0360, A320, 220, Toulouse). Tous les avions ayant même valeur de la rubrique AvNom, n'ont donc plus la même valeur de AvCap. La modification évoquée entraîne une incohérence dans l'ensemble si tous les avions ont forcément même capacité.

Anomalies de stockage

- Les anomalies de stockage interviennent lors des opérations d'insertion, de suppression ou de mise à jour.
  - Peut-on ajouter les couples (A340, 300),(B707,150) c'est-à-dire des valeurs de (AvNom, AvCap), sans précision des autres valeurs?
    - Non, il est nécessaire de donner une valeur de la clé AvNum pour que l'insertion soit possible, or le AvNum n'est pas forcément encore attribué par la compagnie.
  - Peut on supprimer l'avion de numéro 0400?
    - Si on supprime l'avion dont le numéro est 0400, on perd le lien unique concernant le couple (Caravelle, 150).

Problème de reconnexion

- La redondance logique et les anomalies de stockage peuvent être évitées en scindant la relation en deux, mais pas n'importe comment.
- Si on décompose la relation avion en deux relations avion1 et avion2 de schémas respectifs:
  - Avion1(AvNum, AvNom)
  - Avion2(AvNom, AvCap, AvVille)
- La reconstitution de la relation avion à partir des deux relations avion1 et avion2 est incomplète. La décomposition retenue n'est pas inversible.

Preuve

Avion1

AvNum	AvNom
0350	A320
0360	A320
0400	Caravelle
0420	B747
0430	Mercure
0450	Concorde
0500	A320

Avion2

AvNom	AvCap	AvVille
A320	200	Paris
A320	200	Toulouse
Caravelle	150	Nice
B747	500	Lyon
Mercure	100	Nantes
Concorde	150	Paris

Avion

AvNum	AvNom	AvCap	AvVille
0350	A320	200	Paris
0350	**A320**	**200**	**Toulouse***
0360	**A320**	**200**	**Paris***
0360	A320	200	Toulouse
0400	Caravelle	150	Nice
0420	B747	500	Lyon
0430	Mercure	100	Nantes
0450	Concorde	150	Paris
0500	A320	200	Paris
0500	**A320**	**200**	**Toulouse***

2. Normalisation

- La normalisation a pour but de supprimer la redondance logique, d'éviter les anomalies de stockage et de résoudre les problèmes de reconnexion.
- Deux approches ont été proposées :
  - la décomposition d'une relation universelle (contenant tous les attributs) et toutes les dépendances inter-attribut,
  - la synthèse, à partir d'un ensemble de dépendances.

Exemple

- Une centrale d'achat achète en gros des produits qu'elle revend au détail.
- Elle achète les produits chez différents fournisseurs et dispose d'un service achat dans lequel travaillent plusieurs acheteurs.
- Chaque acheteur est responsable d'une catégorie de produits (un acheteur pour l'électroménager, un pour l'habillement, un autre pour la droguerie…).
- Le prix de vente des produits est fixé en fonction du produit.
- Le prix d'achat dépend du fournisseur auprès duquel le produit est acheté.
- Les conditions de livraison et de règlement dépendent des fournisseurs et peuvent évoluer dans le temps.

Collecte des données

- Classification des données
  - Élémentaire
    - Quand on ne peut l'inventer
    - Ex: Le nom d'un fournisseur
  - Paramètre
    - Quand la valeur est constante et prévisible
    - Ex: La TVA si tous les produits ont le même taux.
  - Résultante:
    - Quand on peut la calculer à partir d'une autre.
    - Ex: Le prix TTC = le prix HT * TVA
- Type des données
  - Alphabétique A
  - Alphanumérique AN
  - Numérique N
  - Date D
  - Booléenne B

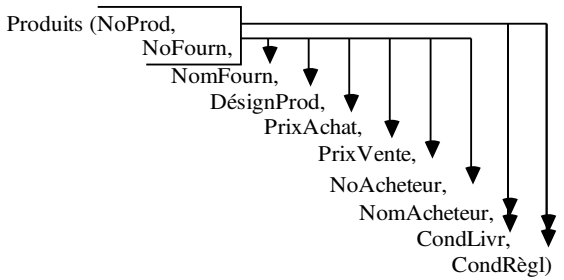
Dictionnaire des données

Donnée	Format	Unité	Type				Règle calcul	Règle intégrité	Document
			Elémentaire	Résultant	Paramètre	Multivaluée			
NomFourn	A		x						Facture
TVA	N	%			x				Textes
PrixVenteHT	N	€	x					>= PrixAchat	Facture
PrixVenteTTC	B	€		x			PrixVente HT*TVA		Facture

- Remarque
  - Ne garder que les données élémentaires pour définir les relations.
  - Supprimer les synonymes et lever les polysémies.
  - Les paramètres seront regroupés dans une table propre ne contenant qu'une seule ligne.
  - Les résultantes sont calculées automatiquement par requêtes SQL.

Dépendance fonctionnelle

- Définition
  - Soit une relation r(R) et deux ensembles d'attributs A et B.
    Les attributs de A déterminent fonctionnellement les attributs de B relativement à R si et seulement si pour deux n-uplets t1 et t2 de r, à chaque fois que t1(A)=t2(A) alors t1A(B)=t2(B).
  - A peut être formé d'un seul attribut ou de plusieurs.
  - On note A->B. A est la source et B le but.



Dépendances fonctionnelles

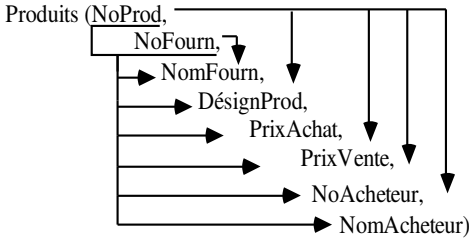
- Il existe plusieurs types de dépendances fonctionnelle :
  - Dépendances fonctionnelles monovaluées (flèche simple) :
    - A un ensemble donné de valeurs sources (*un n° de produit et un n° de fournisseur*) correspond une seule valeur but (*prix de vente*).
  - dépendances fonctionnelles multivaluées (flèche double) :
    - A un ensemble de valeurs sources (*un n° de produit et un n° de fournisseur*) correspondent plusieurs valeurs buts (*conditions de règlement*).
- Exemples :
  - Dépendance fonctionnelle monovaluée.
    - N° étudiant → Nom étudiant
  - Dépendance fonctionnelle multivaluée.
    - N° étudiant → Prénoms étudiant
  - Dépendance non fonctionnelle.
    - Nom étudiant → N° étudiant

Dépendances simples / composées

- Dépendance fonctionnelle simple
  - Cas où la source est composée d'une seule valeur.
    - Ex : NoFour → NomFourn
- Dépendance fonctionnelle à partie gauche composée (DFPGC)
  - Cas où la source est composée de plusieurs valeurs.
    - Ex : (NoProd, NoFour) → NomFourn
- Dépendance fonctionnelle à partie droite composée
  - Faux problème : décomposer en autant de dépendances fonctionnelles à partie droite simple.
    - Ex :
      (NoProd,NoFour) → NomFourn, DesignProd
      (NoProd,NoFour) → NomFourn
      (NoProd,NoFour) → DesignProd

Dépendances élémentaires

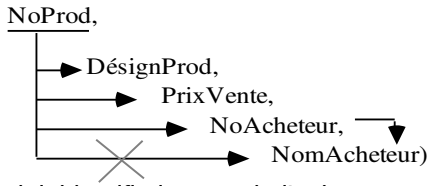
- Une dépendance fonctionnelle est dite élémentaire lorsque chacun des attributs de l'ensemble source est nécessaire pour identifier de façon unique l'attribut but
  - **Toute DF simple est élémentaire**
  - d1→d2 est élémentaire s'il n'existe pas d3 sous-ensemble de d1 tq d3 →d2.



  - Certaines dépendances fonctionnelles (NoProd,NoFourn → NomFourn) ne sont pas élémentaires puisqu'une dépendance fonctionnelle utilisant une partie de l'identifiant identifie le même attribut (NoFourn → NomFourn).

Dépendances directes

- Une dépendance fonctionnelle est dite directe lorsque l'ensemble source est le seul identifiant de l'attribut but.
  - d1 -> d2 est directe s'il n'existe pas d3 qui engendrerait une dépendance fonctionnelle transitive d1 -> d3 tel que d3 -> d2.



  - Le numéro de produit identifie le nom de l'acheteur mais on ne peut ignorer l'existence d'une dépendance fonctionnelle de NoAcheteur vers NomAcheteur.
  - On peut même estimer que la dépendance
    NoProd -> NomAcheteur
    peut être obtenue au moyen des dépendances non directes
    NoProd -> NoAcheteur et NoAcheteur -> NomAcheteur.

3. Décomposition

- Considérons la relation suivante :
  - Produits(NoProd*, NoFourn*, NomFourn, DésignProd, PrixAchat, PrixVente, NoAcheteur, NomAcheteur, CondLivr, CondRègl)
- Supposons que la centrale possède à son catalogue 1000 produits, qu'il y ait en moyenne 3 fournisseurs par produit, 50 fournisseurs au total et 5 acheteurs.
  - Dans ces conditions, nous aurions 3000 lignes pour cette table (1000 produits, 3 fournisseurs par produits). La désignation d'un produit serait répétée en moyenne 3 fois, celle d'un fournisseur plusieurs dizaines de fois, ….
- Pour éviter cette redondance, il faut structurer les informations en plusieurs relations entre lesquelles seront établis des liens = décomposition en formes normales.

Première Forme Normale

- **Une relation (table) est en première forme normale**
  - **Si tous ses attributs sont élémentaires (non décomposables)**
  - **Si elle admet au moins une clé.**
- On élimine les dépendances multivaluées en scindant la table en deux tables de façon à isoler d'une part les attributs en dépendance fonctionnelle et d'autre part les attributs en dépendance multivaluée.
  - La table suivante n'est pas conforme à la 1e forme normale :
    - Produits(NoProd*, NoFourn*, NomFourn, DésignProd, PrixAchat, PrixVente, NoAcheteur, NomAcheteur, CondLivr, CondRègl)
  - On obtient alors les deux tables :
    - Produit(NoProd*, NoFourn*, NomFourn, DésignProd, PrixAchat, PrixVente, NoAcheteur, NomAcheteur)
    - Condition(NoFourn, CondLivr, CondRègl)
  - La table Produit est en première forme normale.

Deuxième Forme Normale

- **Une table est en deuxième forme normale si**
  - **Si elle est en première forme normale**
  - **Si chacun des attributs non membres de la clé est en dépendance fonctionnelle élémentaire vis-à-vis de la clé.**
- On élimine les dépendances non élémentaires en scindant la table en plusieurs tables de façon à conserver d'une part les dépendances déjà élémentaires et d'autre part en créant de nouvelles tables pour les autres dépendances fonctionnelles.
  - La table suivante n'est pas conforme à la 2e forme normale
    - Produit(NoProd*, NoFourn*, NomFourn, DésignProd, PrixAchat, PrixVente, NoAcheteur, NomAcheteur)
  - On obtient alors les tables :
    - Prix(NoProd*, NoFourn*, PrixAchat)
    - Produit(NoProd, DésignProd, PrixVente, NoAcheteur, NomAcheteur)
    - Fournisseur(NoFourn, NomFourn)
  - Les tables Prix, Produit et Fournisseur sont donc en 2e forme normale.

Troisième Forme Normale

- **Une table est en troisième forme normale si**
  - **Si elle est en deuxième forme normale**
  - **Si toutes les dépendances fonctionnelles liant la clé aux autres attributs sont des dépendances directes.**
- On élimine les dépendances non directes en scindant la table Produit de façon à éliminer d'une part la dépendance déductible par transitivité et à introduire d'autre part, la dépendance permettant de déduire la transitivité.
  - La table suivante n'est pas conforme à la 3e forme normale
    - Produit(NoProd, DésignProd, PrixVente, NoAcheteur, NomAcheteur)
  - On obtient alors les tables:
    - Produit(NoProd, DésignProd, PrixVente, NoAcheteur)
    - Acheteur(NoAcheteur, NomAcheteur)
  - Les quatre tables Produit, Prix, Acheteur et Fournisseur sont en 3e forme normale; leur normalisation est achevée.

Forme Normale de Boyce-Codd

- **Une table est en forme normale de Boyce-Codd si**
  - **Si elle est en troisième forme normale**
  - **Si il n'existe aucune dépendance fonctionnelle d'un attribut non clé de la table vers un attribut composant cette clé.**
- Considérons la table suivante:

  Adresse(Ville, Rue, CodePostal)
  - Cette table est en troisième forme normale (un code postal pour (Ville et Rue)), mais elle n'est pas en forme normale de Boyce-Codd, car Ville (attribut membre de la clé) dépend fonctionnellement de CodePostal (attribut non membre de la clé). Notons que ce n'est pas le cas de la rue: un même code postal ne détermine pas une seule rue mais plusieurs.
  - L'anomalie peut être éliminée en scindant la table en deux:

    CodeVille(CodePostal, Ville)

    CodeRue(CodePostal, Rue)

Quatrième et Cinquième Formes Normales

- **Une table est en quatrième forme normale**
  - **Si elle ne contient qu'une dépendance multivaluée.**
- **Exemple**
  - La table qui contient deux dépendances multivaluées

    Condition(NoFourn, CondLivr, CondRègl)
  - devient donc:

    Livraison(NoFourn, CondLivr)

    Règlement(NoFourn, CondRègl)
  - Ces deux tables sont en quatrième forme normale.
- **S'il y avait eu une dépendance multivaluée entre CondLivr et CondRègl, nous aurions ajouté une autre table et parlé de cinquième forme normale.**

Résultat

- La table initiale non normalisée :
  - Produits(NoProd*, NoFourn*, NomFourn, DésignProd, PrixAchat, PrixVente, NoAcheteur, NomAcheteur, CondLivr, CondRègl)
- La base de données enfin normalisée est donc:

  Prix(NoProd*, NoFourn*, PrixAchat)

  Produit(NoProd, DésignProd, PrixVente, NoAcheteur*)

  Acheteur(NoAcheteur, NomAcheteur)

  Fournisseur(NoFourn, NomFourn)

  Livraison(NoFourn*, CondLivr)

  Règlement(NoFourn*, CondRègl)

4. Synthèse: La méthode Merise

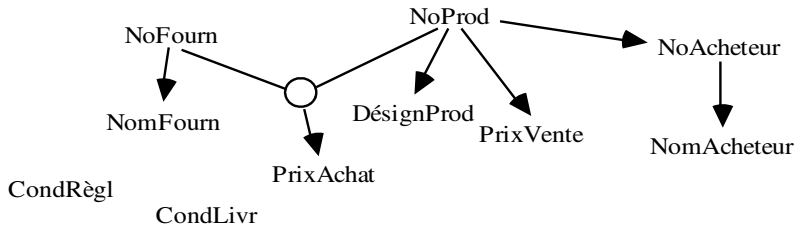
- Merise est une méthode qui gère le cycle de vie d'une base de données. Fait l'hypothèse que l'on eut séparer les traitements des données sur lesquelles portent les traitements.
  - Modèle des Traitements
  - Modèle des Données
- Modèle des Données :
  - La méthode consiste à élaborer un **modèle conceptuel des données** dont on peut déduire un **modèle logique des données**.
  - Le modèle conceptuel fait abstraction des contraintes d'organisation et des contraintes techniques. **Il est basé sur le modèle entité-association.**
  - Le modèle logique tient compte du modèle conceptuel retenu pour structurer les données.
- **Merise garantie que toutes les tables sont en cinquième forme normale**

Méthode de conception: Merise

1. Dictionnaire des données
2. Graphe des dépendances fonctionnelles
3. Conception du modèle conceptuel (entité-association)
   1. Détermination des entités
   2. Détermination des associations
   3. Ventilation des données
   4. Aménagements éventuels
   5. Vérification du modèle conceptuel
4. Conception du modèle logique

Graphe des dépendances fonctionnelles

- Les données du domaine et les dépendances qui les lient peuvent être représentées par un graphe dit **graphe des dépendances fonctionnelles**.
- Toutes les données élémentaires doivent apparaître sur ce graphe mais seules les **dépendances fonctionnelles élémentaires** et **directes** y sont représentées.



Passage du dictionnaire aux dépendances

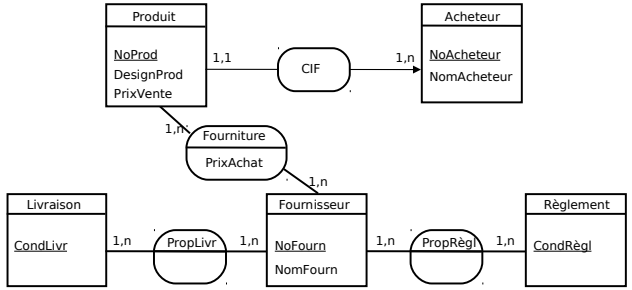
- Recherche des DF simples
  - Il n'y a pas de démarche. Simplement, il faut commencer par les plus évidentes.
- Recherche des DF à partie gauche composée
  1. Construire l'ensemble (S) des sources de DF simples,
  2. Construire l'ensemble (N) des données non concernées comme source ou comme but d'un DF simple,
  3. Rechercher des DFPGC dont la source est constituée de données de l'ensemble (S) ou (N) et le but est une donnée de (N),
  4. Compléter en recherchant des DFPGC dont la source est constituée de données de l'ensemble (S) ou (N) et le but est une donnée de (S),
  5. Eliminer toutes les DF non élémentaires ni directes.

Modèle entité-association

- Le modèle entité-association est un modèle conceptuel des données. Il propose une représentation des données qui s'articule autour de trois concepts :
  - la **propriété** (ou rubrique) est une donnée élémentaire perçue dans le domaine à modéliser. Elle peut concerner une entité ou une association.
    Ex: le numéro de produit, le nom d'acheteur, le prix d'achat
  - l'**entité** est un concept, un objet ou un acteur du domaine qui a une existence propre et conforme aux règles de gestion du domaine.
    Ex: produit et fournisseur.
  - l'**association** rend compte d'un lien entre entités. Elle n'a pas d'existence propre mais peut être porteuse de propriétés.
    Ex: lien entre produit et fournisseur.

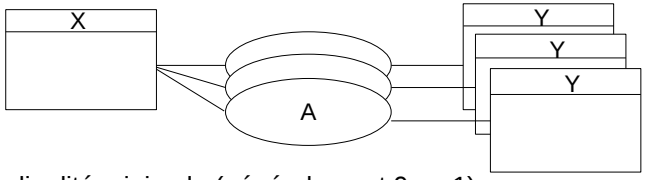
Représentation

- Entité : un rectangle surmonté de son nom.
- Propriété : attribut d'une entité ou d'une association.
- Association :
  - un ovale séparé en deux parties par un trait horizontal
    - La partie supérieure pour le nom, la partie inférieure pour les propriétés.
  - des traits relient l'ovale aux entités participant à l'association. Chaque trait est annoté par sa cardinalité.

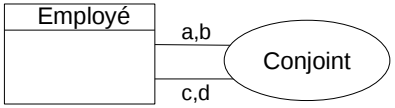


Cardinalités d'une association

- Le nombre minimum et maximum de fois qu'une occurrence de l'entité participe aux occurrences de l'association (≠ UML).



  - m cardinalité minimale (généralement 0 ou 1)
  - M cardinalité maximale (généralement 1 ou n)
- Importance des cardinalités
  - Exemple de l'association *Employé a pour conjoint* (-> diverses sociétés)

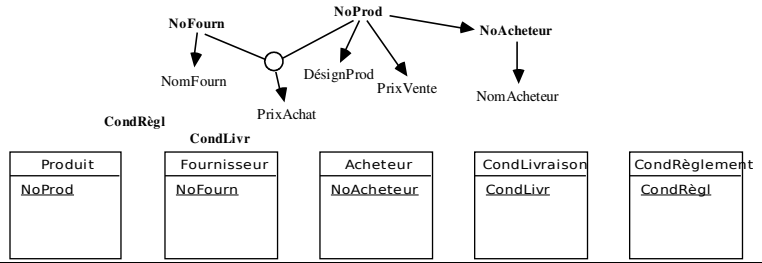


a- Détermination des entités

- Les données sources sont des propriétés identifiantes d'entités. Elles présentent au moins une des quatre caractéristiques suivantes:
  - être source de dépendance fonctionnelle simple (entre deux données)
  - être membre d'une source de dépendance fonctionnelle à partie gauche composée
  - être but de plusieurs dépendances fonctionnelles bien que n'étant pas source de dépendance fonctionnelle
  - être une donnée isolée, concernée par aucune dépendance fonctionnelle.
- Chaque source détermine une entité dont elle est l'identifiant.

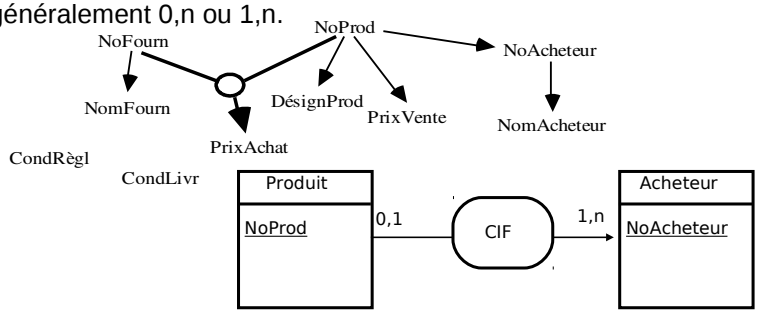
Exemple

- Les sources déterminent donc ici cinq entités :
  - Produit, Fournisseur, Acheteur, CondLivraison et CondRèglement.
  - Produit est l'objet du domaine.
  - Acheteur et Fournisseur en sont les acteurs. Tous ont une existence propre. Il est impossible de décrire le domaine sans les évoquer.
  - De la même façon, on ne peut pas décrire le domaine sans parler des conditions de livraison et des conditions de règlement. Ces deux notions, bien que non matérielles, n'en sont pas moins des réalités : elles ont donc une existence propre conforme aux règles de gestion : des entités.



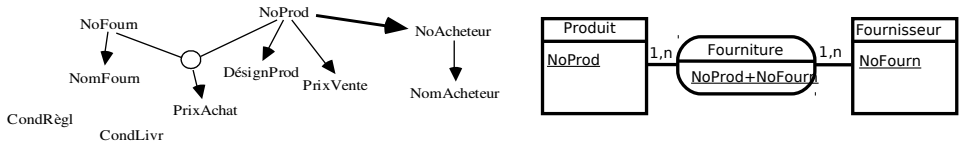
b- Détermination des associations binaires

- Une dépendance fonctionnelle dont le but est une source d'une autre dépendance ou but de plusieurs (i.e., ∈ S) donne une Contrainte d'Intégrité Fonctionnelle (CIF), association binaire à cardinalité maximum unitaire (0,1 ou 1,1).
  - L'association ne porte pas de nom et ne possède pas de propriétés.
  - La cardinalité 0,1 ou 1,1 sera placée du côté de l'entité où se trouve la source de la dépendance fonctionnelle. L'autre cardinalité sera généralement 0,n ou 1,n.



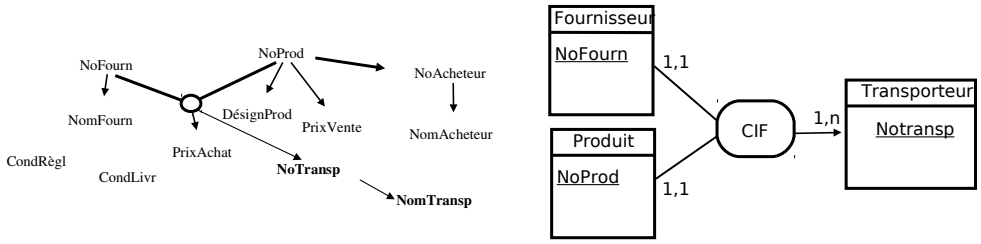
Détermination des associations n-aires

- Une dépendance fonctionnelle à partie gauche composée dont le but n'est pas une source donne une association n-aire entre les entités identifiées par les données constituant la source.
  - Les cardinalités seront généralement 0,n ou 1,n.
  - L'identifiant est formé de la concaténation des identifiants des sources.
    - Exemple: Une association n-aire relie donc les entités Produit et Fournisseur. Ici, un produit fait l'objet d'au moins une fourniture et un fournisseur propose au moins une fourniture. Les cardinalités min sont donc égales à 1. La cardinalité min unitaire implique que seuls les fournisseurs avec lesquels travaille la centrale d'achat seront consignés dans la banque de données. Pour indiquer la présence éventuelle de fournisseurs avec lesquels on ne travaille pas, il faudrait fixer à 0 la cardinalité min.



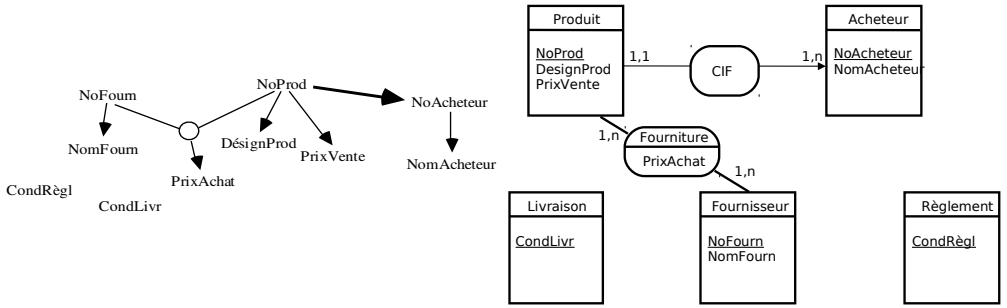
Détermination des CIF multiples

- Chaque dépendance fonctionnelle à partie gauche composée dont le but est une source donne une contrainte d'intégrité fonctionnelle multiple (CIF multiple) qui part des entités identifiées par les données constituant la source de la dépendance et pointe vers l'entité correspondant au but.
  - Les cardinalités sur l'arc pointant vers le but sont généralement 0,n ou 1,n.
  - L'association ne porte généralement pas de nom et ne possède pas de propriétés.



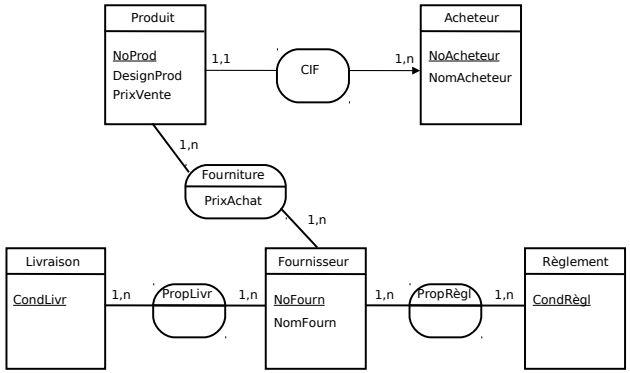
c- Ventilation des données

- Les données buts de dépendances fonctionnelles simples, et qui ne sont pas des sources, sont des propriétés des entités identifiées par l'origine de ces dépendances.
  - Les données buts de dépendances fonctionnelles à partie gauche composée, et qui ne sont pas des sources, sont des propriétés des associations auxquelles participent les entités identifiées par les données buts de ces dépendances.



d- Aménagement du modèle conceptuel

- Les données isolées, buts d'aucune dépendance fonctionnelle conduisent à des entités isolées pour lesquelles il peut être nécessaire de créer une association vers une entité non isolée du modèle conceptuel de données.



e- Vérification du modèle conceptuel de données

1. Il doit exister un identifiant pour chaque entité. Ce qui n'a pas d'identifiant est donc une association.
2. Pour chaque occurrence d'une entité, chaque propriété ne peut prendre qu'une seule valeur.
3. Toutes les propriétés doivent être élémentaires, c'est-à-dire non décomposables (1ère forme normale).
4. Toutes les propriétés autres que l'identifiant doivent dépendre pleinement et directement de l'identifiant. C'est-à-dire qu'elles doivent dépendre de tout l'identifiant et non d'une partie de cet identifiant (2ème forme normale); et non par l'intermédiaire d'une ou plusieurs autres propriétés (3ème forme normale).
5. A chaque occurrence d'une association correspond une seule occurrence de chaque entité participant à l'association (2 occurrences d'une entité ne peuvent participer à une même occurrence d'association. Pour une occurrence d'association, il n'y a pas de participation optionnelle d'une entité).
6. Pour chaque occurrence d'une association, il ne peut exister une seule valeur pour chaque propriété de l'association.
7. Toutes les propriétés d'une association doivent dépendre pleinement de l'identifiant de l'association, c'est-à-dire de la concaténation des identifiants des entités participant à l'association.

e- Vérification du modèle conceptuel de données

1. Il doit exister un identifiant pour chaque entité. Ce qui n'a pas d'identifiant est donc une association.
2. Pour chaque occurrence d'une entité, chaque propriété ne peut prendre qu'une seule valeur.
3. Toutes les propriétés doivent être élémentaires, c'est-à-dire non décomposables (1ère forme normale).
4. Toutes les propriétés autres que l'identifiant doivent dépendre pleinement et directement de l'identifiant. C'est-à-dire qu'elles doivent dépendre de tout l'identifiant et non d'une partie de cet identifiant (2ème forme normale); et non par l'intermédiaire d'une ou plusieurs autres propriétés (3ème forme normale).
5. A chaque occurrence d'une association correspond une seule occurrence de chaque entité participant à l'association (2 occurrences d'une entité ne peuvent participer à une même occurrence d'association. Pour une occurrence d'association, il n'y a pas de participation optionnelle d'une entité).
6. Pour chaque occurrence d'une association, il ne peut exister une seule valeur pour chaque propriété de l'association.
7. Toutes les propriétés d'une association doivent dépendre pleinement de l'identifiant de l'association, c'est-à-dire de la concaténation des identifiants des entités participant à l'association.

Le modèle logique

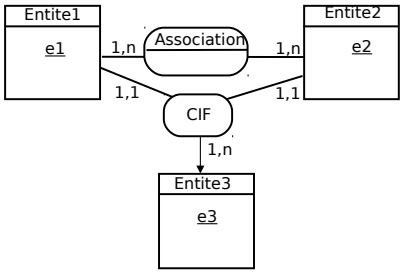
- Le modèle logique décrit les schémas relationnels de la base de données (les tables).
- Il peut être directement implémenté dans une système de base de données relationnel.
- Avec un système de gestion de fichiers, il faut en plus décrire le modèle physique qui est la description des fichiers correspondants.
- Toutes les données de type **paramètres** sont regroupées dans une table unique.

Passage du modèle conceptuel au modèle logique

- Entités:
  - Chaque entité devient une relation. L'identifiant de l'entité devient la clé de la relation. Les propriétés de l'entité deviennent des attributs de la relation.
- CIF simples:
  - L'identifiant de l'entité du côté (0,n) ou (1,n) devient attribut de la relation qui résulte de l'entité côté (0,1) ou (1,1). Cet attribut est une clé étrangère. Les propriétés de l'association deviennent des attributs de la relation qui résulte de l'entité côté (0,1) ou (1,1).
- Associations n-aires:
  - Chaque association devient une relation. L'identifiant de chaque association devient clé de la relation. Les propriétés de l'association deviennent attributs de la relation.

Passage du modèle conceptuel au modèle logique

- Pour les CIF multiples deux cas:
  - Accompagnée d'une association entre les mêmes entités sources
    - L'identifiant de l'entité du côté (0,n) ou (1,n) devient attribut de la relation qui résulte de l'association. Cet attribut est une clé étrangère.
  - Non accompagnée d'une association
    - Créer une association de toute pièce entre les entités sources.
    - L'identifiant de l'entité du côté (0,n) ou (1,n) devient attribut de la nouvelle relation. Cet attribut est une clé étrangère.



Prise en compte des cardinalités

- Les cardinalités induisent des contraintes sur les attributs des tables.
  - NOT NULL pour les clés étrangères.
  - Des vérifications par check….

Application à l'exemple

- Les cinq entités donnent les relations:

  Produit(NoProd, DésignProd, PrixVente)

  Acheteur(NoAcheteur, NomAcheteur)

  Fournisseur(NoFourn, NomFourn)

  CondLivraison(CondLivr)

  CondRèglement(CondRègl)
- Les trois associations du type (1,n) vers (1,n) donnent les relations:

  Fourniture(NoProd*, NoFourn*, PrixAchat)

  PropLivraison(NoFourn, CondLivr)

  PropRèglement(NoFourn, CondRègl)
- L'association du type (1,1) vers (1,n) conduit à ajouter le numéro d'acheteur dans la relation Produit:

  Produit(NoProd, DésignProd, PrixVente, NoAcheteur*)

Application à l'exemple (2)

  - Les relations CondLivraison et CondRèglement ne contiennent pas d'autres attributs que leur clé. Puisqu'on retrouve ces clés dans les relations PropLivraison et PropRèglement, les relations CondLivraison et CondRèglement n'apportent rien. On peut donc s'en passer.

- Ce qui donne la base de données:

  Produit(NoProd, DésignProd, PrixVente, NoAcheteur*)
  Acheteur(NoAcheteur, NomAcheteur)
  Fournisseur(NoFourn, NomFourn)
  Fourniture(NoProd*, NoFourn*, PrixAchat)
  PropLivraison(NoFourn, CondLivr)
  PropRèglement(NoFourn, CondRègl)

Application à l'exemple (3)

  - On peut cependant s'interroger sur la disparition des relations CondLivraison et CondRèglement. Si les propriétés CondLivr et CondRègl sont des chaînes de caractères correspondant aux libellés desdites conditions, il vaut sans doute mieux éviter de les dupliquer et faire en sorte qu'elles ne soient pas identifiant des entités correspondantes. On préfère alors attribuer à chaque entité de ce type un code numérique, beaucoup moins volumineux qui en sera l'identifiant.

- On obtiendrait alors une base de données de schéma logique :

  Produit(NoProd, DésignProd, PrixVente, NoAcheteur)*
  Acheteur(NoAcheteur, NomAcheteur)
  Fournisseur(NoFourn, NomFourn)
  CondLivraison(CodeLivr, LibLivr)
  CondRèglement(CodeRègl, LibRègl)
  Fourniture(NoProd*, NoFourn*, PrixAchat)
  PropLivraison(NoFourn*, CodeLivr)
  PropRèglement(NoFourn*, CodeRègl)

Documents contractuels

- Dictionnaire des données
  - Le type des données
  - Les unités
  - Le domaine d'intégrité
- MCD
  - Les entités
  - Les associations
- Eventuellement, MLD s'il a été aménagé pour l'efficacité

- Ces 2 (ou 3) documents sont nécessaires et suffisants pour implémenter la base de données, et toute implémentation définira exactement les mêmes tables.
- **Il ne peut y avoir de base de données sans ces documents.**