IIT
INSTITUT INTERNATIONAL TECHNOLOGIE
Université Nord Américaine privée

Ingénierie des Bases de Données

Mohamed Amine CHAÂBANE
Mohamedamine.chaabane@isaas.usf.tn

Introduction au modèle Entité - Association

I. Définition des concepts de base

- *1. "Objet-type"* (ou, par abus de langage, "Objet") : C'est une classe d'entités manipulée par l'organisme, dotée d'une existence propre, identifiable et ayant un intérêt particulier pour l'organisme.
- Exemples : PRODUIT ; FOURNISSEUR ; EMPLOYE.

I. Définition des concepts de base

2. *"Relation-type"* ou *"Association-type"* (ou ", par abus de langage, "Relation" ou "Association") : Une association est définie sur une collection de N objets-types. L'existence d'une association est conditionnée par celle des objets qui la composent. Elle représente un prédicat (vrai ou faux) qui lie ces objets. Elle peut être porteuse de données.

   Exemples : FOURNITURE entre les objets-types PRODUIT et FOURNISSEUR .

I. Définition des concepts de base

3. *"Propriété-type"* (ou, par abus de langage, "Propriété") : C'est le plus petit élément d'information qui a un sens en lui même et qui caractérise soit un objet, soit une association.

   Exemple : PRIX-ACHAT caractérise l'association FOURNITURE.

   Remarque : À chaque objet-type, relation-type et propriété-type correspond un certain nombre d'occurrences.

I. Définition des concepts de base

4. *Identifiant*

- ***Identifiant d'un objet*** : C'est une propriété particulière de l'objet, choisie de telle manière qu'à chaque valeur (à chaque occurrence) prise par cette propriété correspond une et une seule occurrence de cet objet.

Exemple : NUM-EMPL est l'identifiant de l'objet EMPLOYE.

Remarque : Un identifiant peut être formé, dans des cas particuliers, de deux ou de plusieurs propriétés.

- ***Identifiant d'une Association*** : c'est la concaténation des identifiants des objets qui participent à la relation.

Exemple : (NUM-PROD, NUM-FOUR) est l'identifiant de l'association FOURNITURE.

I. Définition des concepts de base

5. *Dimension d'une Association :* C'est le nombre d'objets-types participant à cette association-type.

Exemple : La relation FOURNITURE est de dimension 2.

On peut avoir :

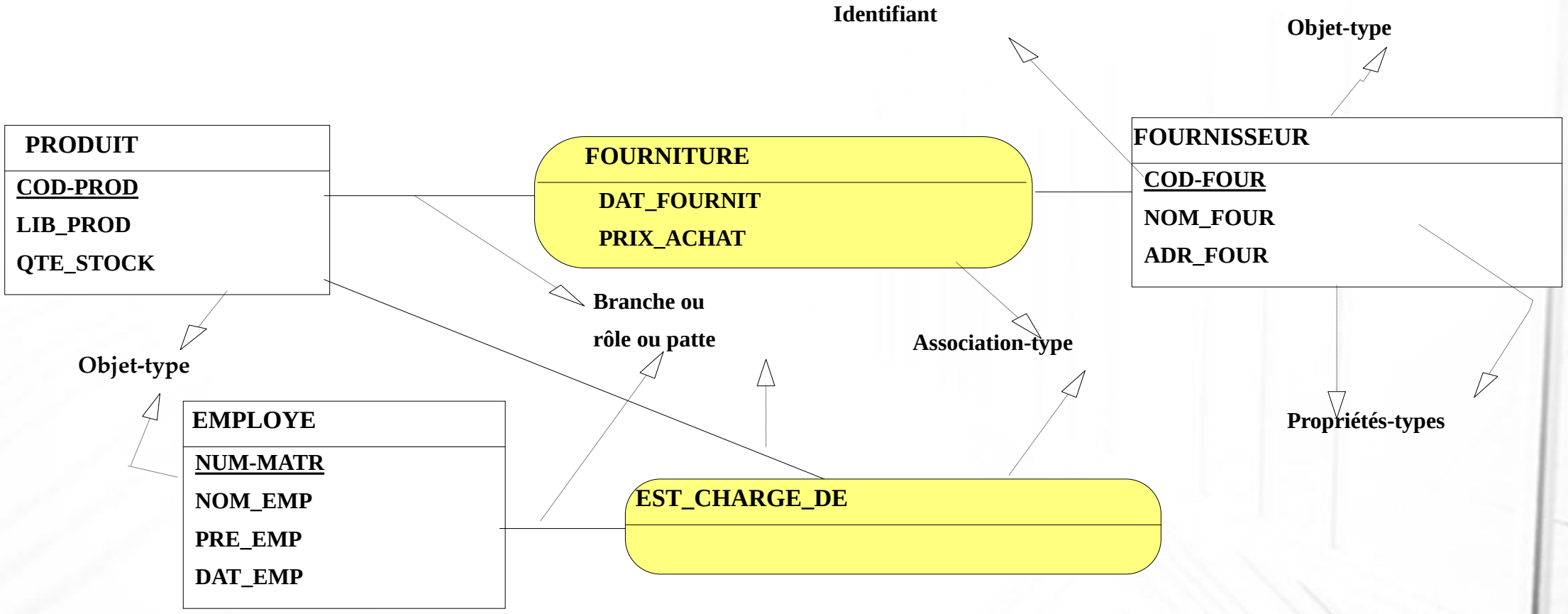
- des relations binaires, c'est-à-dire de dimension 2 ;
- des relations ternaires, c'est-à-dire de dimension 3 ;
- des relations quaternaires, c'est-à-dire de dimension 4 ; ...
- des relations n-aires, c'est-à-dire de dimension n.

I. Définition des concepts de base

6- ***Modèle conceptuel des données (MCD) :*** Un MCD, ou schéma conceptuel des données, est une représentation fidèle des données du champ de l'étude et des liens sémantiques entre ces données, construite, en premier lieu, à l'aide des concepts précédents.

II. Formalisme graphique

Le formalisme graphique utilisé dans MERISE est illustré par l'exemple suivant :



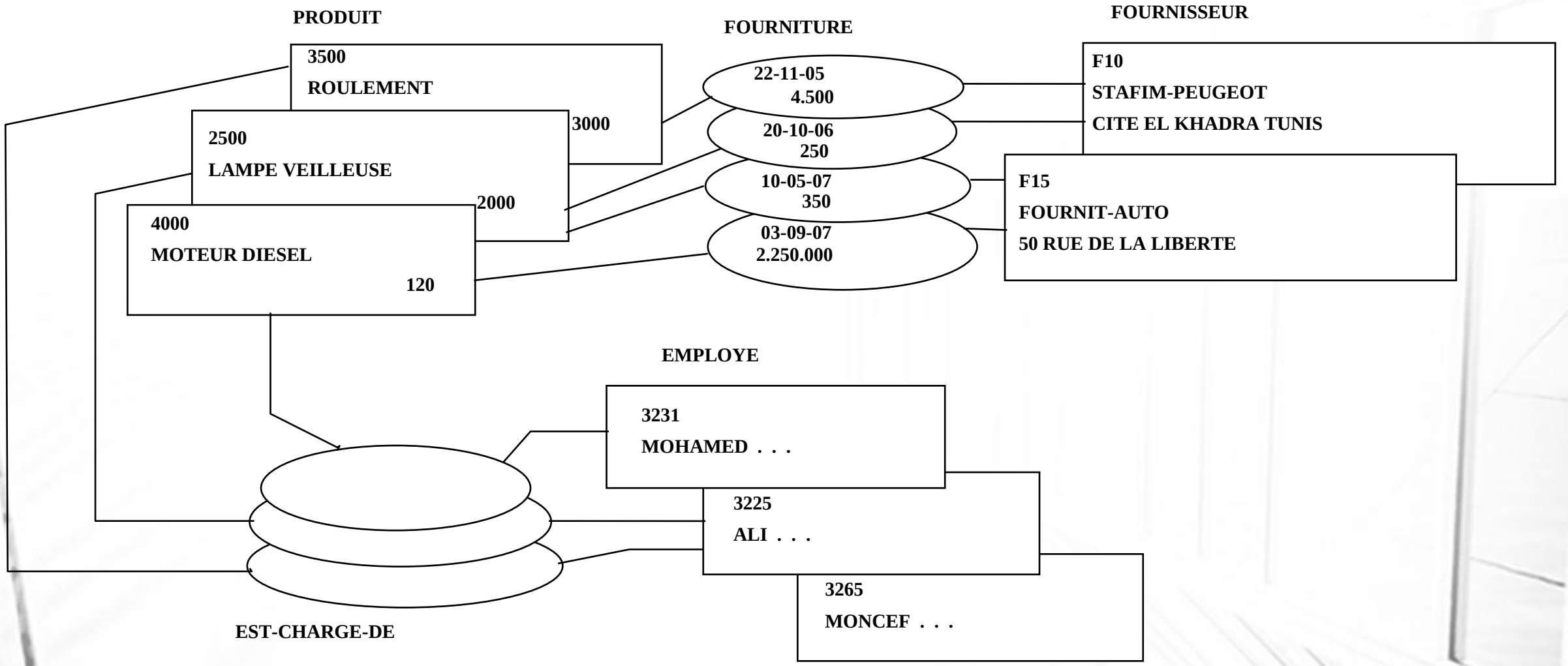
Exemple de MCD

II. Formalisme graphique

Remarques :

- Les identifiants sont soulignés.
- La liste détaillée des propriétés peut être présentée ailleurs.
- Les propriétés peuvent être numérotées. Les numéros peuvent alors remplacer les codes des propriétés peu importantes dans le schéma conceptuel.
- Le modèle est un schéma conceptuel des données. Il est la représentation des types d'occurrences et non pas des occurrences elles-mêmes qui peuvent exister réellement. La représentation de la situation réelle des occurrences ne peut être assurée que par un système de gestion des données. Tout de même, il est parfois intéressant de représenter un extrait de cette situation, conformément au modèle construit, afin d'illustrer et valider ce modèle, c'est-à-dire de vérifier qu'il trace fidèlement la réalité. Cet extrait est appelé "schéma d'occurrences".

II. Formalisme graphique



Exemple d'un schéma d'occurrences

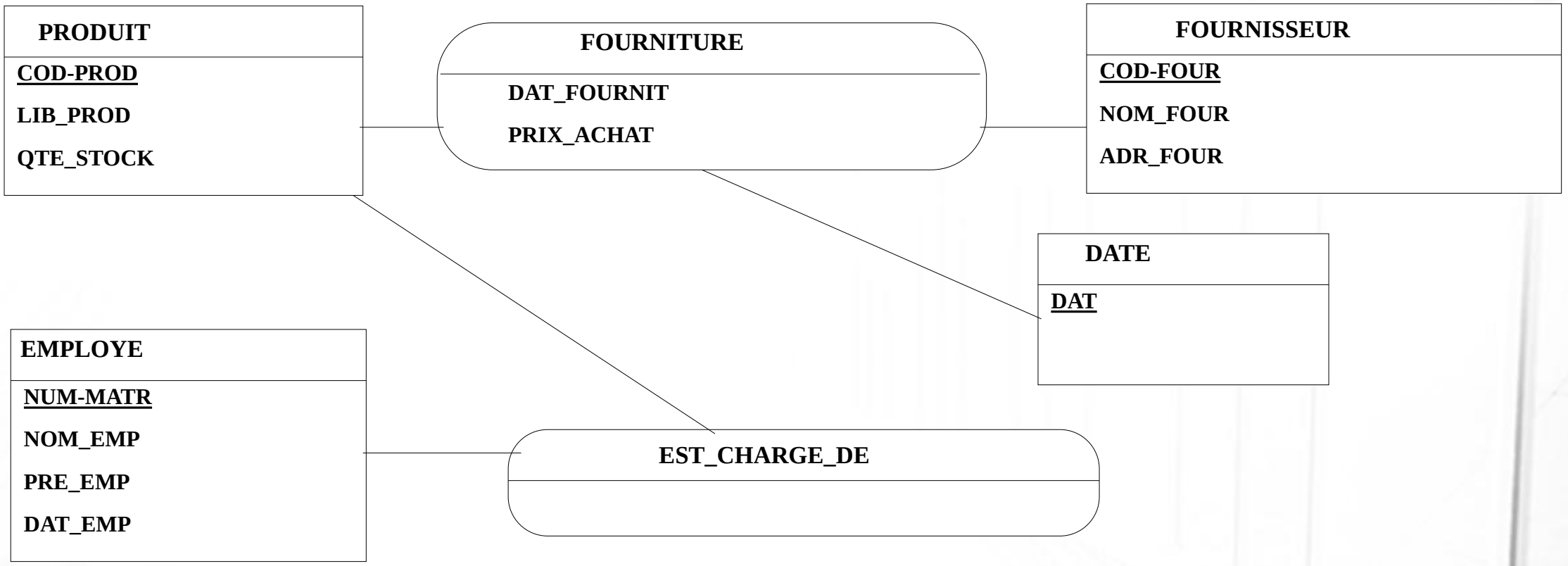
II. Formalisme graphique

Remarque :

Il est impossible d'ajouter une nouvelle occurrence entre le fournisseur "F10" et le produit "3500", par exemple. En effet, on aurait dans ce cas deux occurrences de FOURNITURE qui ont la même valeur d'identifiant (COD-PROD + COD-FOUR), ce qui est interdit. Que faire alors si dans la réalité on a besoin de représenter une telle occurrence ?

La solution consiste ici à agréger la propriété DAT-FOURNIT de l'association FOURNITURE pour former un nouvel objet, l'objet DATE, qui doit participer alors à l'association FOURNITURE.

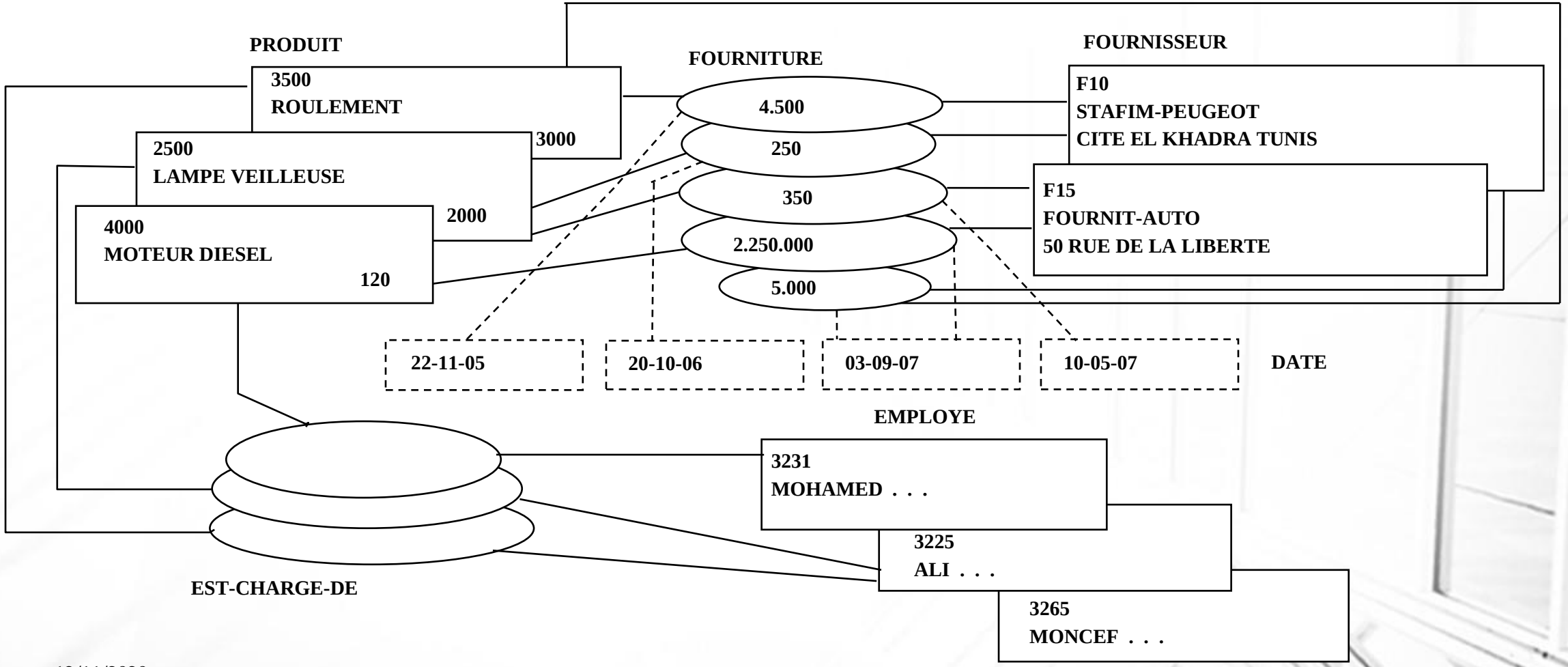
II. Formalisme graphique



Identifiant de FOURNITURE = COD-PROD + COD-FOUR + DAT

II. Formalisme graphique

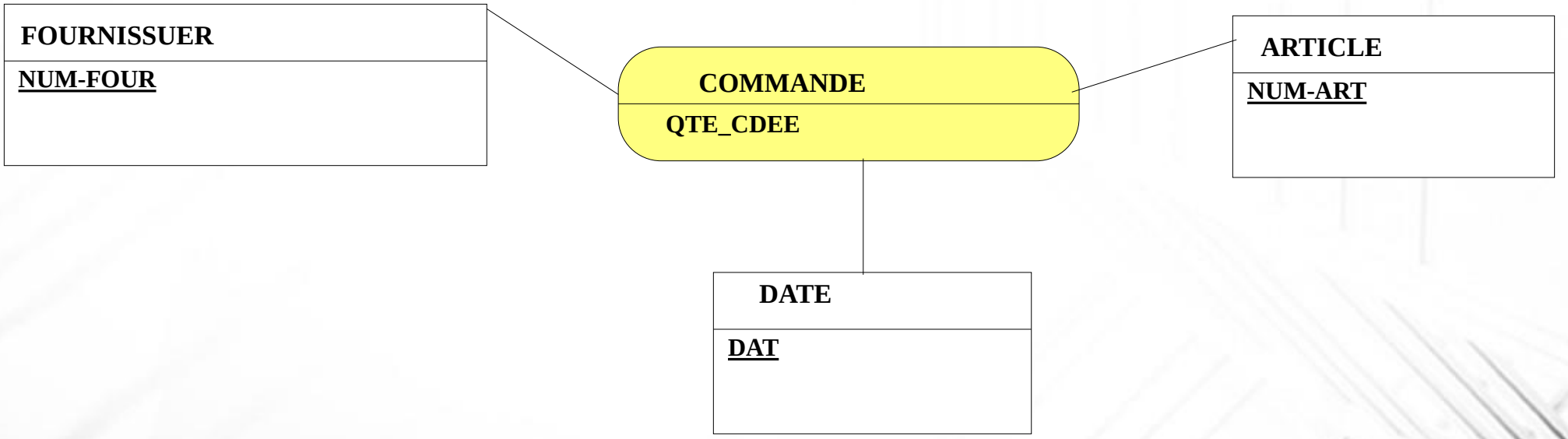
Le schéma d'occurrences devient alors comme suit :



II. Formalisme graphique

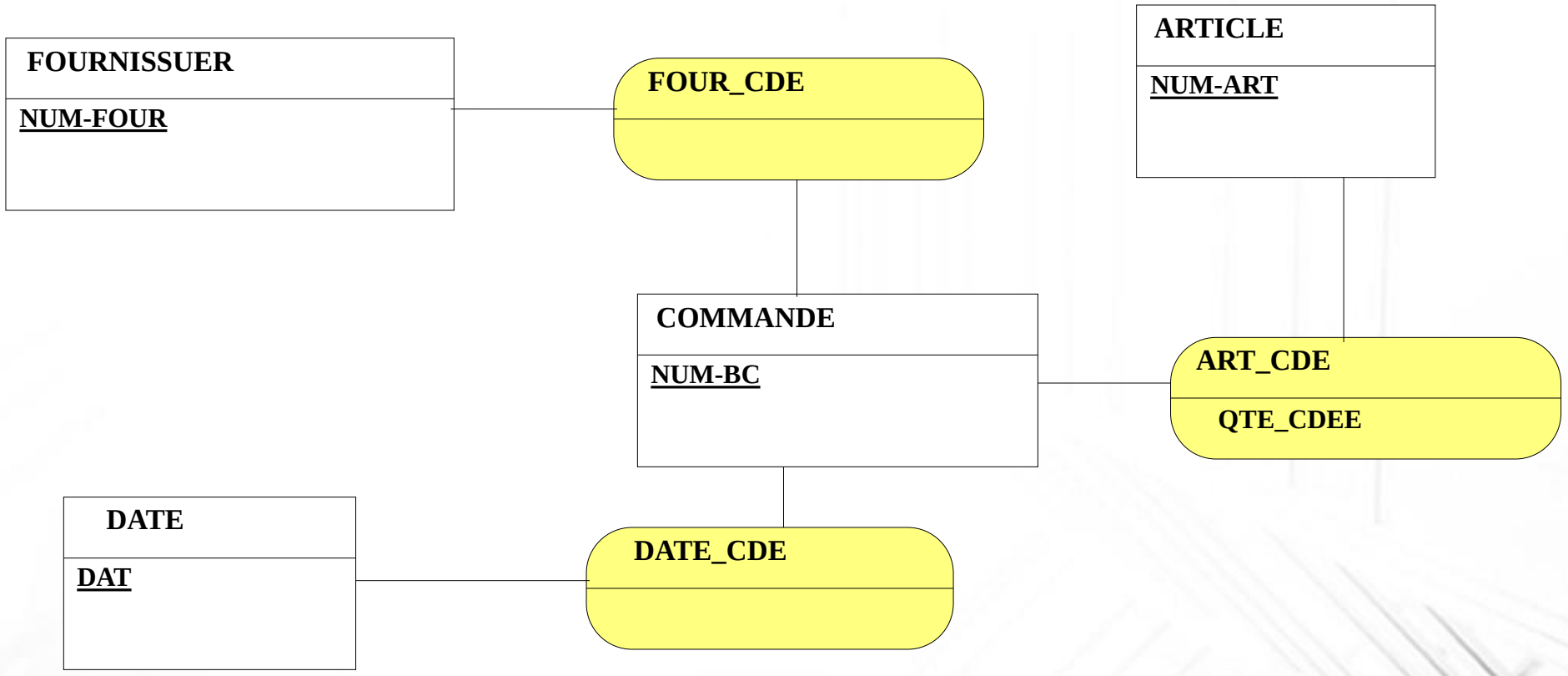
Remarque :

Il est habituel, dans certains cas, de regrouper plusieurs articles commandés chez le même fournisseur dans un même bon de commande. Le modèle ci-dessous représenté ne correspond pas donc à certaines réalités du fait qu'il provoque des redondances au niveau de la branche vers DATE et au niveau de la branche vers FOURNISSEUR.



II. Formalisme graphique

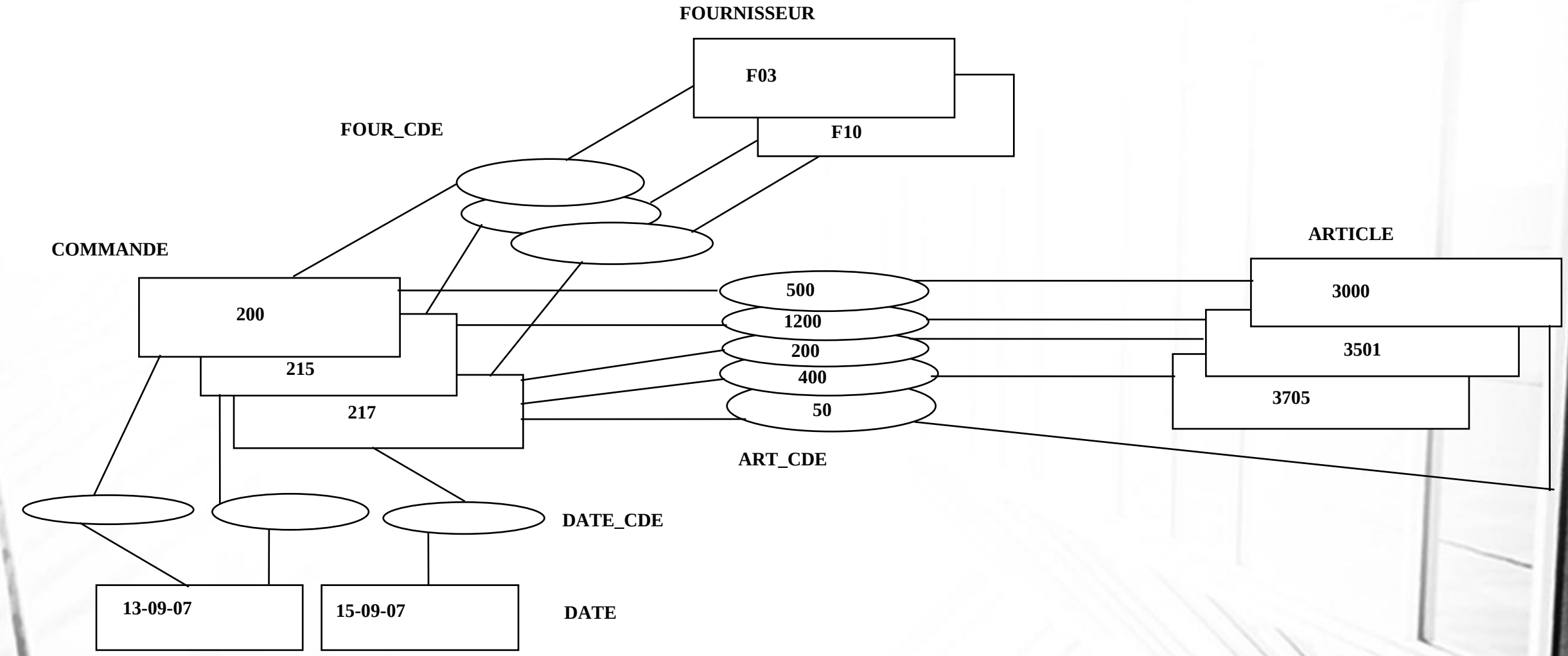
Pour corriger ce modèle, il est possible de créer un objet COMMANDE identifié conformément à une codification adéquate. Le schéma devient comme suit :



II. Formalisme graphique

Ce deuxième modèle donne une meilleure présentation du regroupement des articles commandés chez le même fournisseur et à une même date, tel que montré par le schéma d'occurrences suivant :

II. Formalisme graphique



III. Cardinalités d'une relation

Les cardinalités d'une relation expriment la participation, à cette relation, de chacun des objets qui la composent. Pour chaque objet, cette participation peut être :

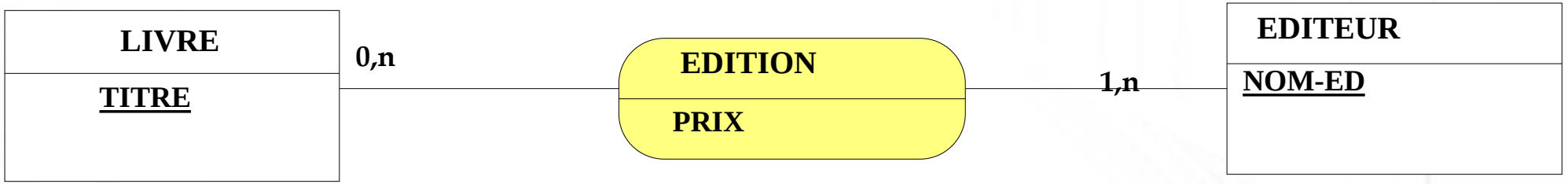
- TOTALE : c'est-à-dire que chaque occurrence de l'objet participe à la relation.
- PARTIELLE : c'est-à-dire que certaines occurrences de l'objet ne participent pas à la relation.

III. Cardinalités d'une relation

Définition

Les cardinalités d'une relation indiquent, pour chaque couple Objet-Relation, les nombres minimum et maximum d'occurrences de la relation pouvant exister pour une même occurrence de l'objet.

Exemple :



III. Cardinalités d'une relation

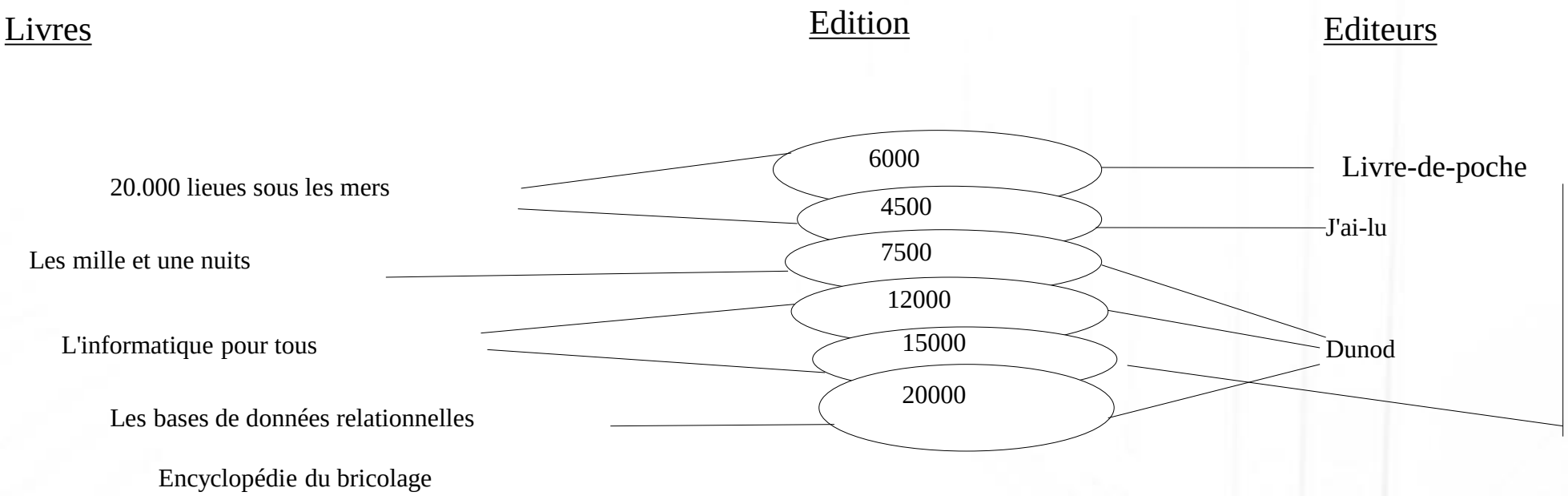
- *Sur la branche LIVRE-EDITION*
  - La cardinalité minimale "0" indique le fait que l'on peut avoir des livres non édités ; il s'agit de manuscrits par exemple.
  - La cardinalité maximale "n" indique le fait qu'un livre peut avoir plusieurs éditions.

III. Cardinalités d'une relation

- *Sur la branche EDITEUR-EDITION*
  - La cardinalité minimale "1" indique le fait que le système d'information ne prend en charge une occurrence EDITEUR que lorsque cette occurrence participe au moins une fois dans l'association EDITION, c'est-à-dire lorsque la bibliothèque dispose d'au moins d'un livre provenant de cet éditeur.
  - La cardinalité maximale "n" indique le fait qu'une occurrence EDITEUR peut participer plusieurs fois dans la relation EDITION.

III. Cardinalités d'une relation

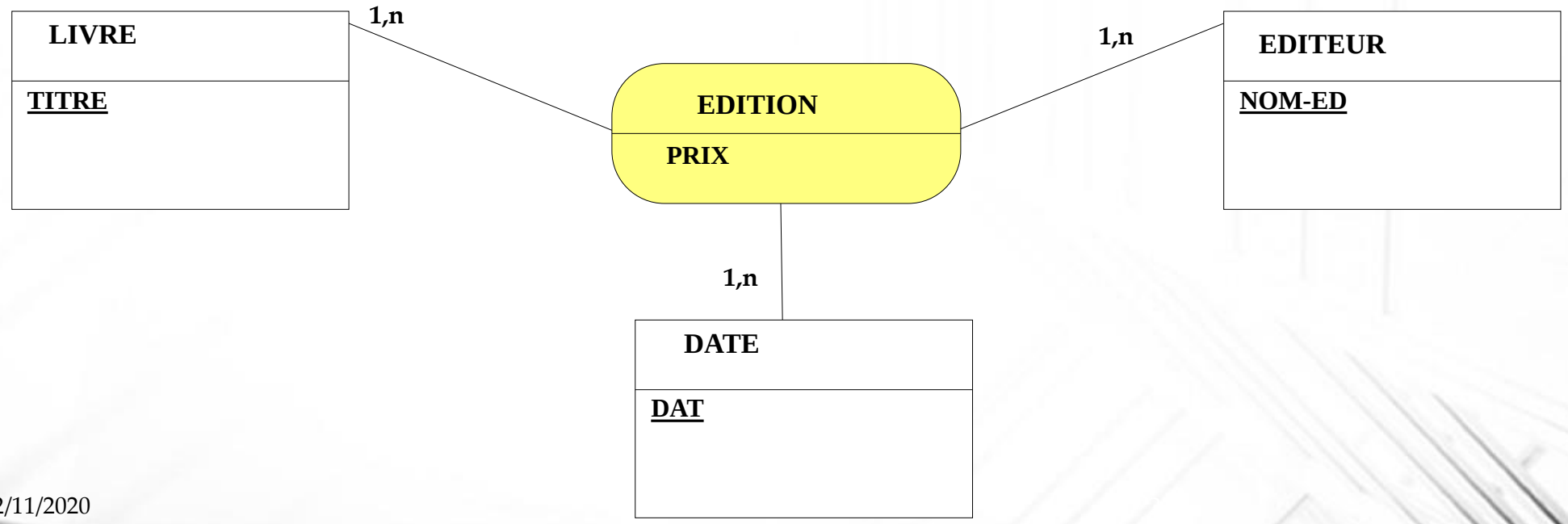
Un exemple de schéma d'occurrences est ci-dessous présenté.



III. Cardinalités d'une relation

Par contre cette démarche ne peut pas être appliquée au cas de relations reliant plus de deux objets (relations n-aires). Le seul moyen pour déterminer les cardinalités de ces relations consiste à considérer chaque couple Objet-Relation à part.

Exemple :

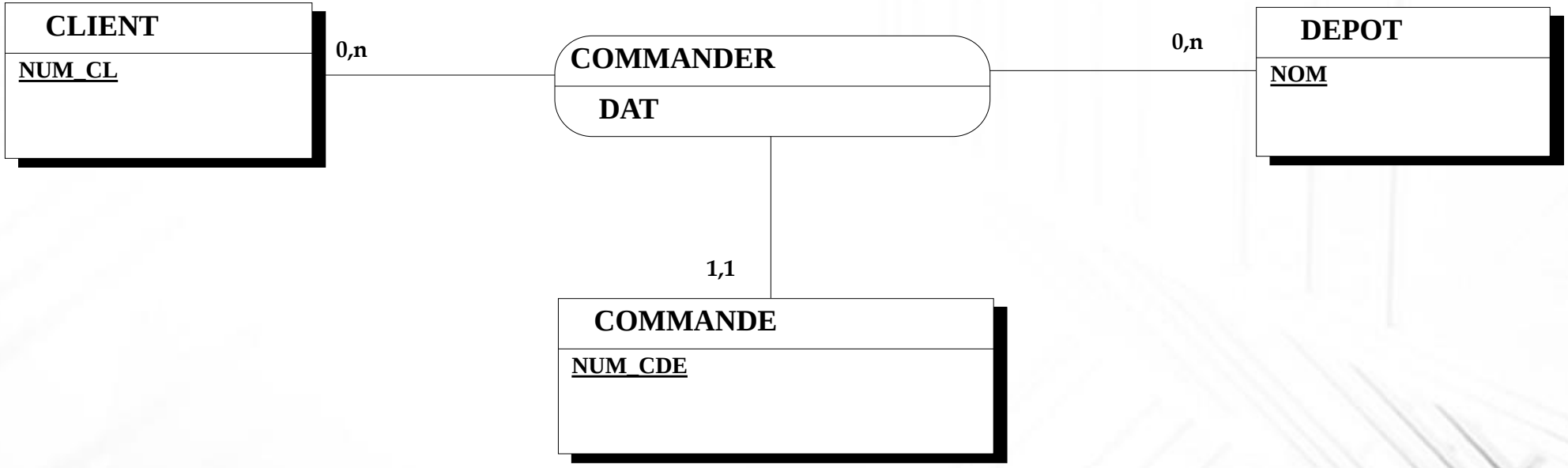


III. Cardinalités d'une relation

Remarque :

Dans le cas d'une cardinalité de type 1-1, il faut placer les propriétés de la relation dans l'objet concerné.

- Exemple :



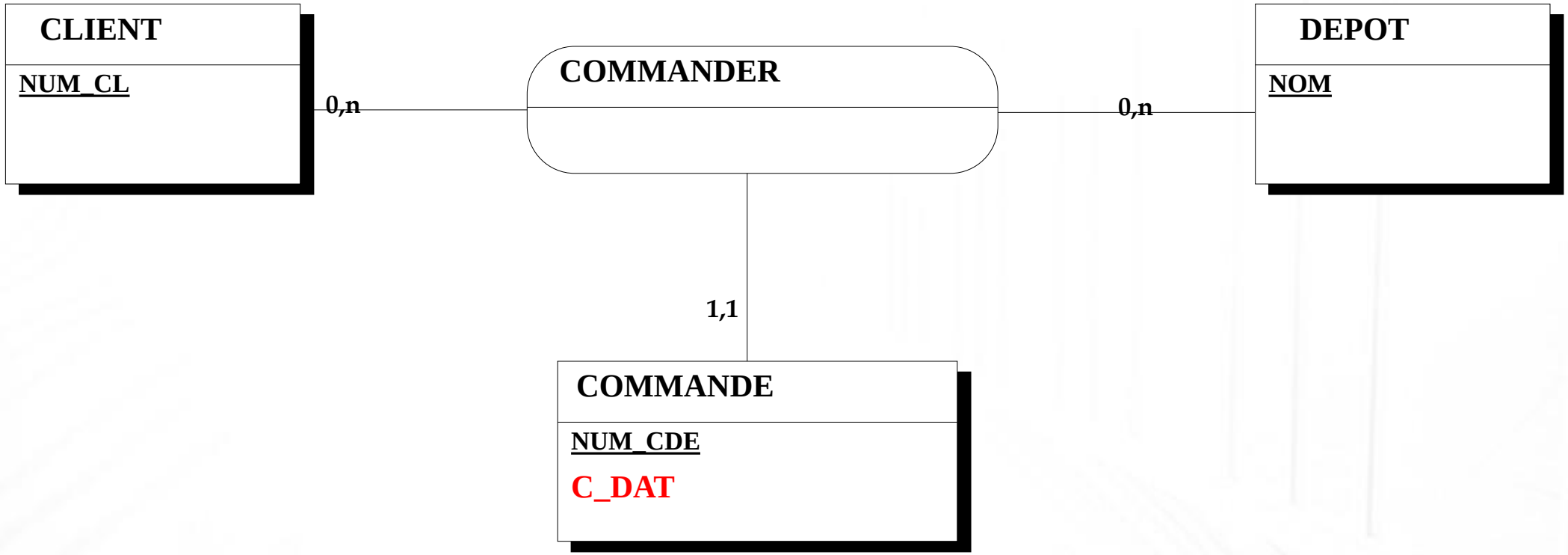
III. Cardinalités d'une relation

- Les cardinalités "1,1" sont expliquées par l'application de la règle de gestion qui dit : "une commande émane d'un et d'un seul client et doit être satisfaite à travers un et un seul dépôt". A chaque occurrence de l'objet COMMANDE correspond une et une seule occurrence de l'association COMMANDER.

- La date doit alors être une propriété de l'objet COMMANDE ; on dit que l'on fait migrer la date de la relation vers l'objet.

III. Cardinalités d'une relation

Le modèle précédent doit être ainsi corrigé :



Les règles de passage vers un modèle relationnel

I. Règles de passage MCD-MLD brut

Règle 1 : Cas d'un individu non vide

Tout individu non vide (comportant au moins une propriété, autre que son identifiant) se transforme en une relation au sens relationnel du terme :

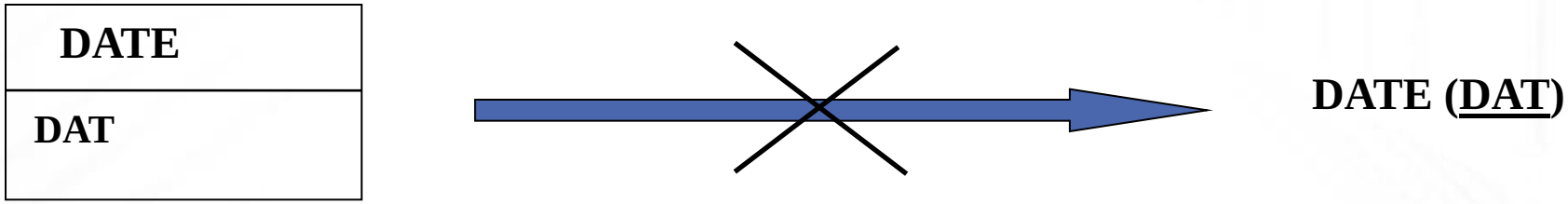
- L'identifiant de l'individu devient la clé de la relation
- Les propriétés de l'individu deviennent les attributs de la relation.

I. Règles de passage MCD-MLD brut

Règle 2 : Cas d'un individu vide

Tout individu dont la liste des propriétés se limite à son identifiant n'aura pas de correspondant dans le modèle logique, sauf exception.

- Exemple : On ne définit pas une relation pour l'objet DATE.

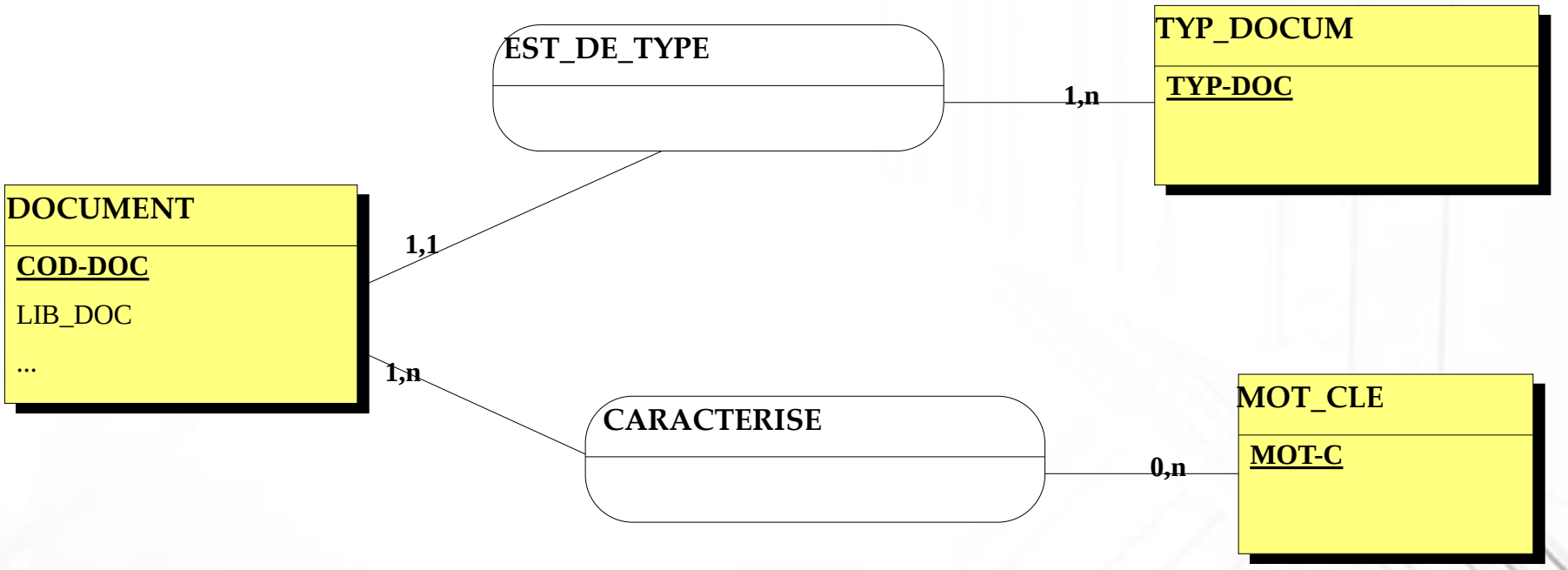


I. Règles de passage MCD-MLD brut

Exception de la Règle 2

La règle 2 n'est pas applicable lorsqu'il est nécessaire de constituer un répertoire des données de l'objet vide et que la projection des autres relations sur l'attribut concerné ne donne pas ce répertoire.

Exemple :



I. Règles de passage MCD-MLD brut

- DOCUMENT (CODE-DOC, LIB-DOC, ...)
- MOT-CLE (MOT-C)
- MOT-CLE ≠ $\Pi_{\text{MOT-C}}$ CARACTERISE
- TYPE-DOCUM = $\Pi_{\text{TYP-DOC}}$ EST_DE_TYPE

I. Règles de passage MCD-MLD brut

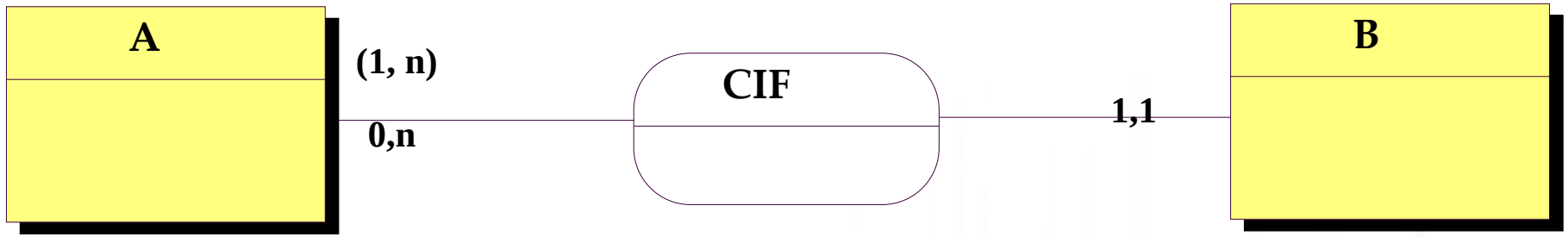
Règle 3 : Cas d'une relation

Toute relation conceptuelle, qui n'est pas une DF (ayant 1,1 comme paire de cardinalité), se transforme en une relation au sens relationnel du terme :

- L'identifiant de la relation conceptuelle devient la clé primaire de la relation relationnelle.
- Chaque attribut de la clé primaire est défini également comme étant une clé étrangère (foreign key), sauf si l'objet-type correspondant n'est pas transformé en une relation.
- Une clé étrangère permet de déclarer une contrainte référentielle.
- Les propriétés de la relation conceptuelle (si elle est porteuse de données) deviennent les attributs de la relation relationnelle.
- Exemple : CARACTERISE (CODE-DOC#, MOT-C #)

I. Règles de passage MCD-MLD brut

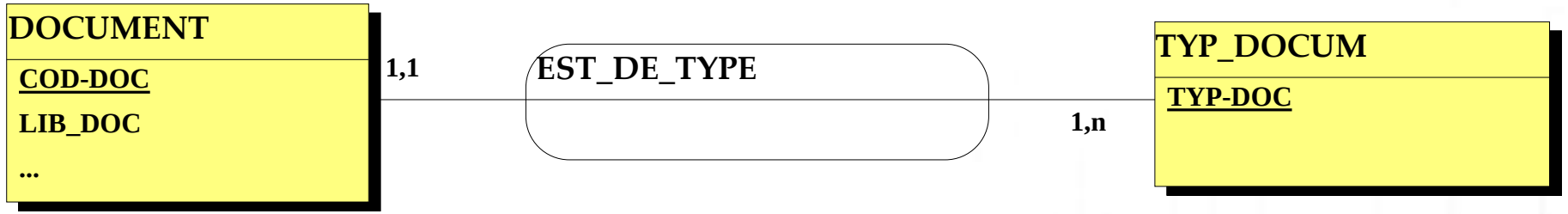
Règle 4 : Cas d'une DF binaire forte



- Les deux individus A et B se transforment conformément à la règle 1 et à la règle 2.
- L'identifiant de A (cible) donne lieu à un attribut clé étrangère dans la relation B (au sens relationnel) lorsque A se transforme en une relation, ou à un attribut simple autrement.
- La DF ne se transforme pas en relation, sauf exception.

I. Règles de passage MCD-MLD brut

Exemple



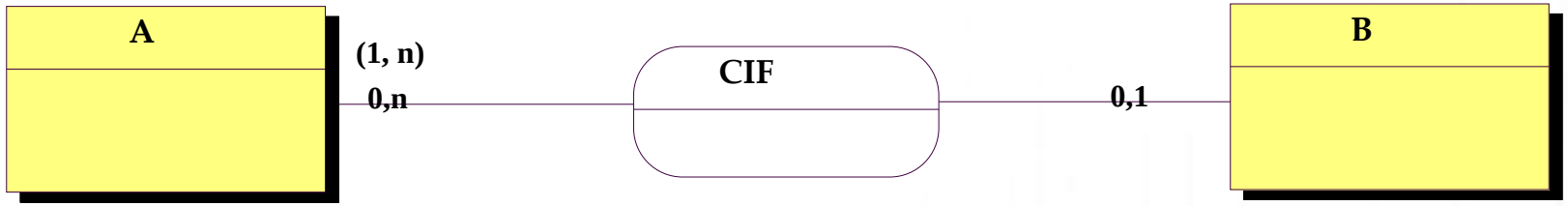
DOCUMENT (COD-DOC, LIB_DOC, TYP-DOC)

Exception à la règle 4 :

Pour une DF binaire dont l'objet source est un objet vide, il faut correspondre une relation, au sens relationnel, soit à la DF, soit à l'objet vide.

I. Règles de passage MCD-MLD brut

Règle 5 : Cas d'une DF binaire faible



- *Solution n°1 :*

Elle consiste à appliquer les mêmes dispositions que celles de la règle 4.

Toutefois, la relation « B » (au sens relationnel du terme) ne sera pas normalisée.

I. Règles de passage MCD-MLD brut

En effet, toutes les occurrences de « B » ne vont pas avoir la même liste d'attributs :

l'attribut correspondant à l'identifiant de « A » et, éventuellement, ceux correspondant aux propriétés de la DF n'existent dans la relation « B » que pour les occurrences qui participent à la DF.

- *Solution n°2 :*

- Les deux individus A et B se transforment conformément à la règle 1 et à la règle 2.

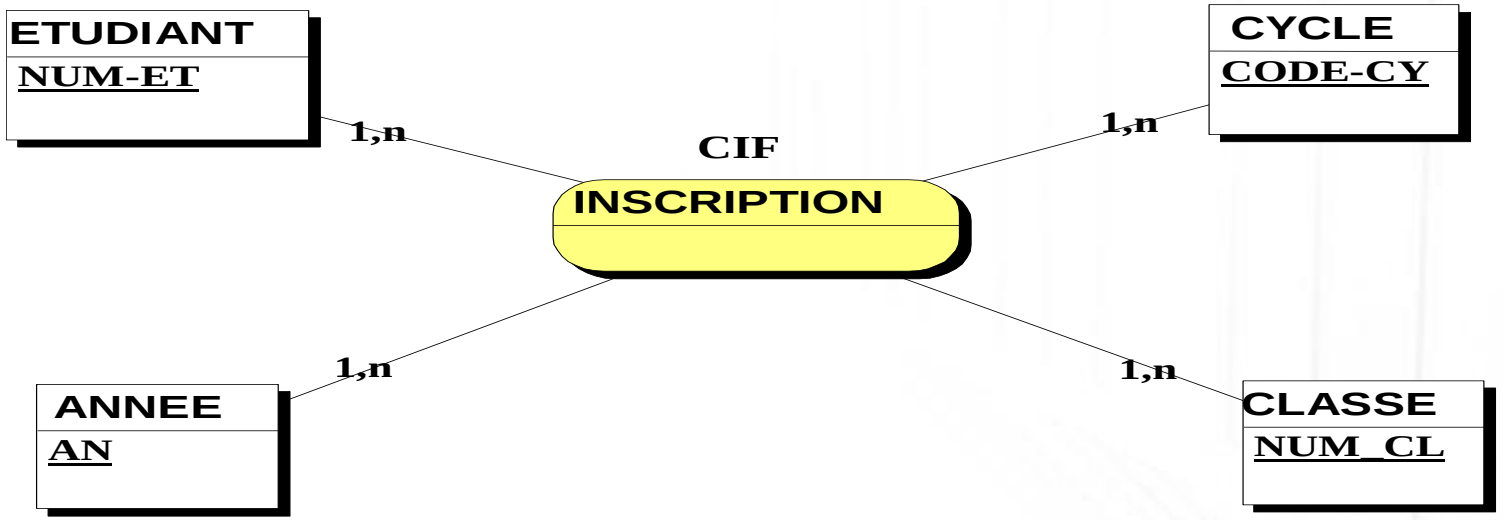
- La DF se transforme conformément à la règle 3.

I. Règles de passage MCD-MLD brut

Remarque :

Les DF non binaires dont la source ne se limite pas à un seul objet se transforment conformément à la règle 3.

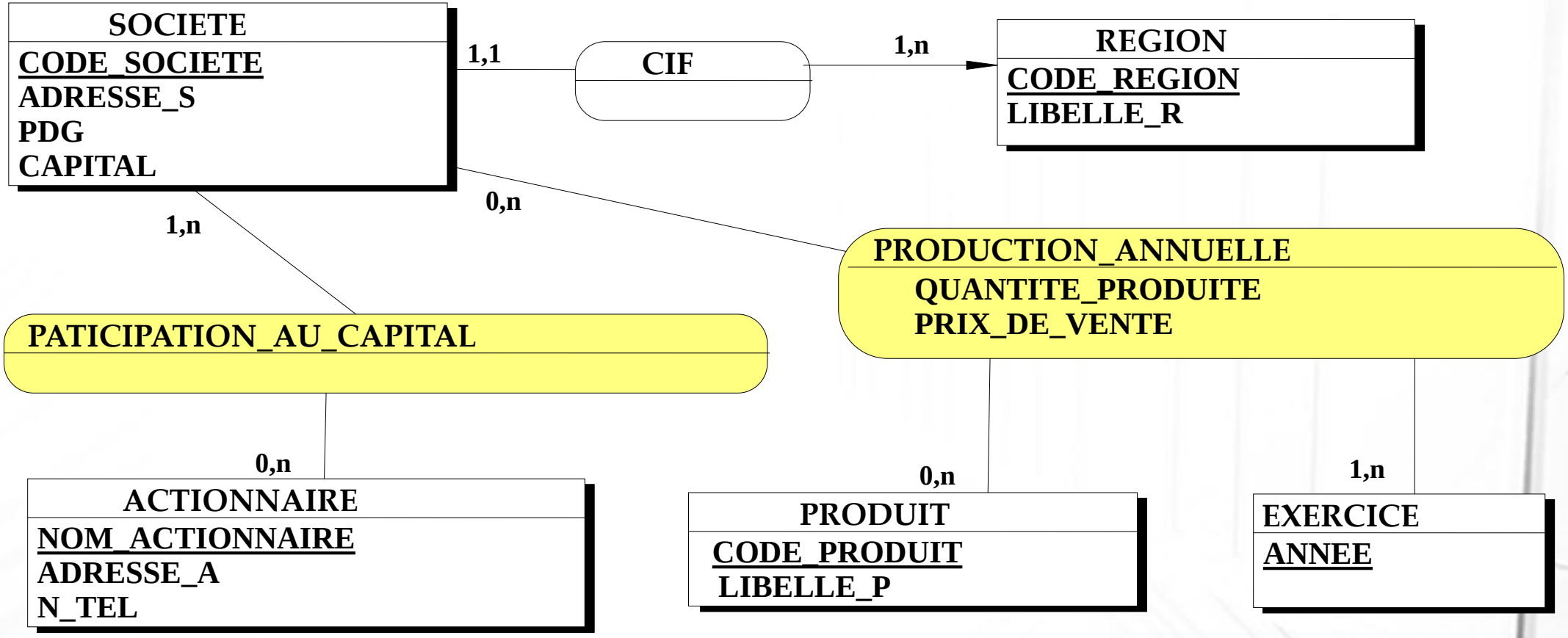
Exemple :



INSCRIPTION (NUM_ET#, AN, COD_CY, NUM_CL)

I. Règles de passage MCD-MLD brut

Exemple Récapitulatif :



I. Règles de passage MCD-MLD brut

SOCIETE (CODE_SOCIETE, ADRESSE_S, PDG, CAPITAL, CODE_REGION #)

REGION (CODE_REGION, LIBELLE_R)

ACTIONNAIRE (NOM_ACTIONNAIRE, ADRESSE_A, N_TEL)

PARTICIPATION_AU_CAPITAL (CODE_SOCIETE#, NOM_ACTIONNAIRE #)

PRODUIT (CODE_PRODUIT, LIBELLE_P)

PRODUCTION_ANNUELLE (ANNEE, CODE_SOCIETE #, CODE_PRODUIT #, QUANTITE_PRODUIT, PRIX_VENTE)

I. Règles de passage MCD-MLD brut

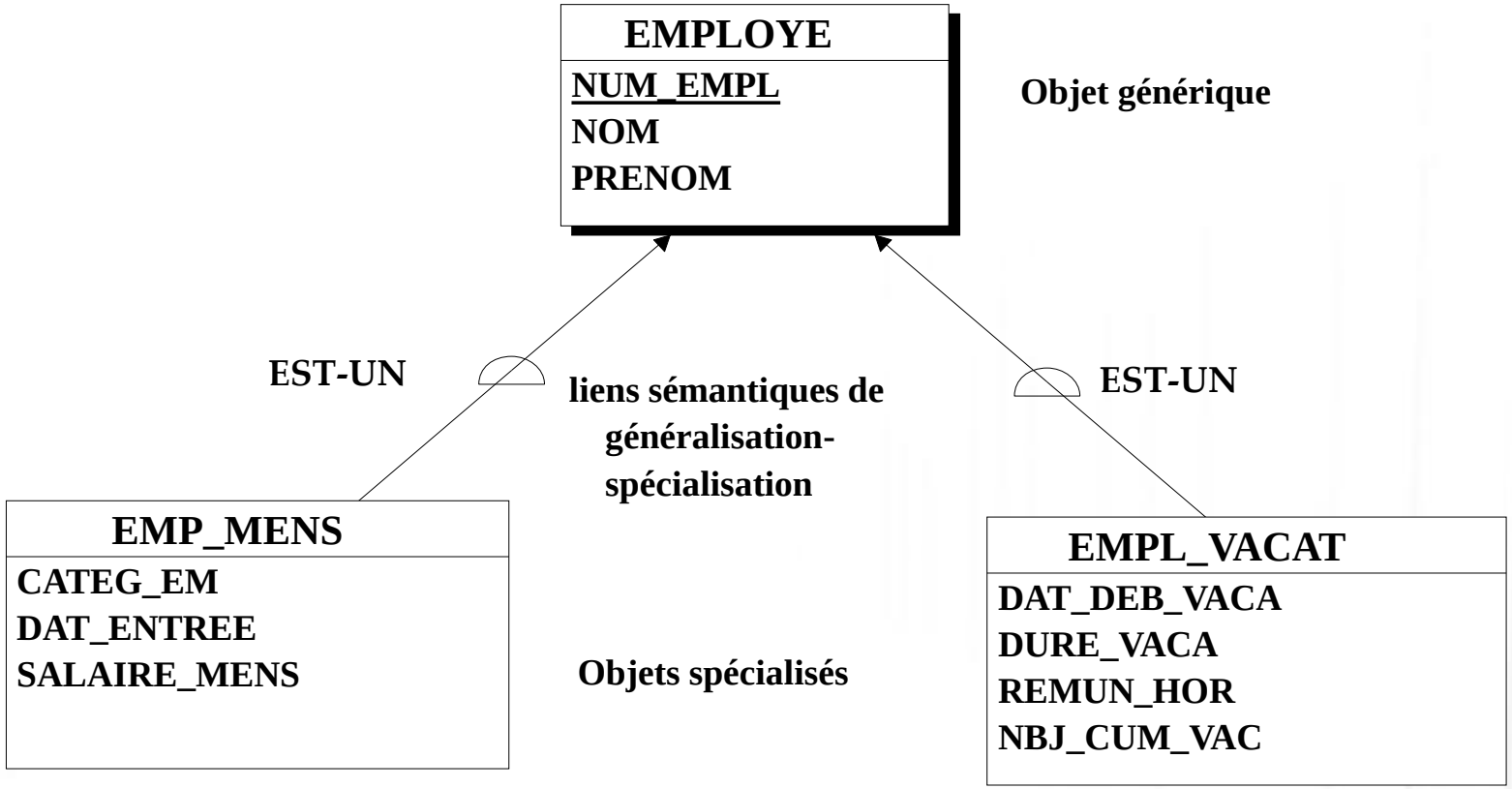
Règle 6 : Cas des généralisations / spécialisations

- Un objet générique se transforme conformément aux règles 1 et 2.

- Un objet spécialisé se transforme en une relation dont la clé primaire est la même que celle de l'objet générique et les attributs sont les transformés des propriétés de l'objet spécialisé. Cette clé primaire est définie aussi comme clé étrangère lorsque l'objet générique se transforme en une relation.

I. Règles de passage MCD-MLD brut

Exemple :

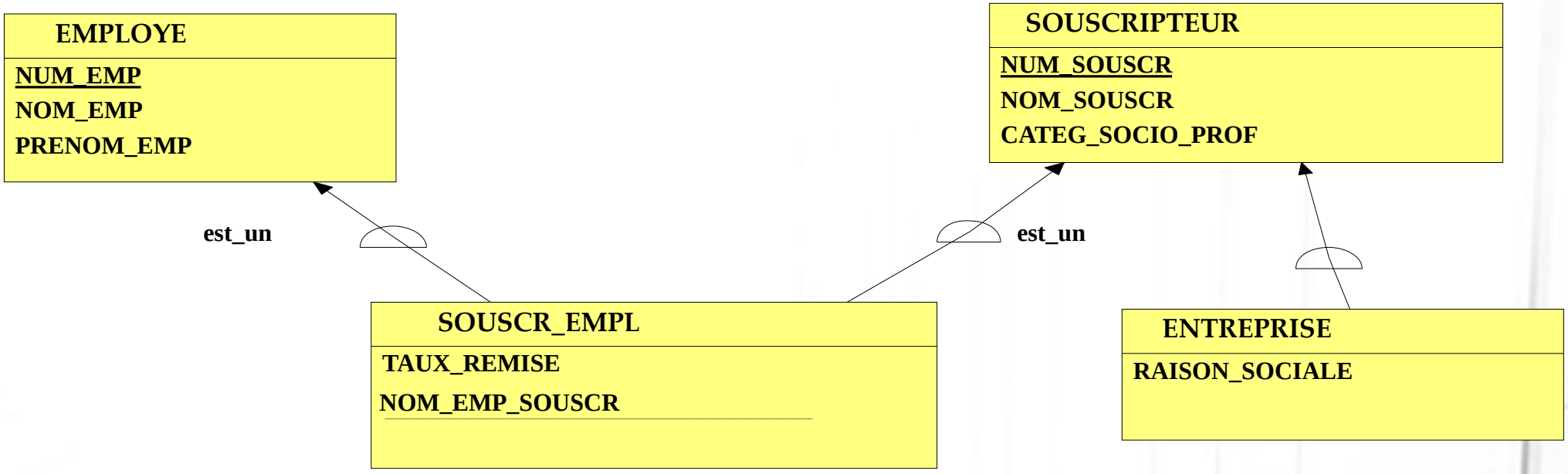


EMPLOYE (NUM_EMPL, NOM, PRENOM)

EMPL_MENS (NUM_EMPL#, CATEG_EM, DAT_ENTREE, SALAIRE_MENS)

I. Règles de passage MCD-MLD

Cas d'une généralisation composée :



EMPLOYE (NUM_EMP, NOM_EMP, PRENOM_EMP)

SOUSCRIPTEUR (NUM_SOUSCR, NOM_SOUSCR, CATEG_SOCIO_PROF)

ENTREPRISE (NUM_SOUSCR, RAISON-SOCIALE, ADR_ENT, SECTEUR_ENT)

SOUSCR_EMP (NUM_EMP#, NUM_SOUSCR #, TAUX_REMISE)

I. Règles de passage MCD-MLD brut

Remarque : La relation SOUSCR_EMP comporte deux clés candidates.

Exceptions :

- Dans le cas où tous les objets spécialisés de l'objet générique sont vides, ces objets ne se transforment pas en relations.
- En effet, il aurait suffit d'ajouter une propriété dans l'objet générique spécifiant le type de chaque occurrence.
- La relation correspondante à cet objet générique comporte alors un attribut indiquant le type de chaque tuple.

I. Règles de passage MCD-MLD brut

Remarque : La relation SOUSCR_EMP comporte deux clés candidates.

Exceptions :

- Dans le cas où tous les objets spécialisés de l'objet générique sont vides, ces objets ne se transforment pas en relations.
- En effet, il aurait suffit d'ajouter une propriété dans l'objet générique spécifiant le type de chaque occurrence.
- La relation correspondante à cet objet générique comporte alors un attribut indiquant le type de chaque tuple.

Introduction aux Bases de Données

Introduction

- **Les entreprises gèrent des volumes de données très grands**
  - Giga, Terra, Péta –octets
  - Numériques, Textuelles, Multi-média (images, films,...)
- **Il faut pouvoir facilement**
  - Archiver les données sur mémoires secondaires permanente
  - Retrouver les données pertinentes à un traitement
  - Mettre à jour les données variant dans le temps
- **Les données sont structurées et identifiées**
  - Données élémentaires ex: Votre salaire, Votre note en BD
  - Données composées ex: Votre CV, vos résultats de l'année
  - Identifiant humain ex: NSS ou machine: P26215

Un peu d'historique

- **Années 60 :**
  - Récipients logique de données ➔ fichiers sur disque
  - Accès séquentiel puis sur clé
  - Lire (Nomf, Article), Ecrire (Nomf, Article)
  - Lire (Nomf, Article, Clé), Ecrire (Nomf, article, Clé)
- **Années 70 :**
  - Avènement des Bases de Données Réseaux (BD)
  - Ensemble de fichiers reliés par des pointeurs
  - Langage d'interrogation par navigation
- **Années 80 :**
  - Avènement des Bases de Données Relationnelles (BDR)
  - Relations entre ensemble de données
  - Langage d'interrogation par assertion logique

Systèmes de fichiers

Comptabilité

Chirurgie

Consultations

Psychiatrie

Caractéristiques

Problèmes

Format des fichiers

Dupont

Symptomes : y
Turlututu : sqj
Symptomes : y
Turlututu : sdd
Analyses : xxx

Dupond

Turlututusqjsk
Symptom: yyyy
Analyses xxxx

Turlututudhjsd
Analyses :xx

Duhpon

Symptomes : yy
Analyses : xxxx

Symptomes : yy

Duipont

Turlututu : sq

Symptomyyyy
Analysesxxxx

Turlututudhjsd

Caractéristiques

Plusieurs applications

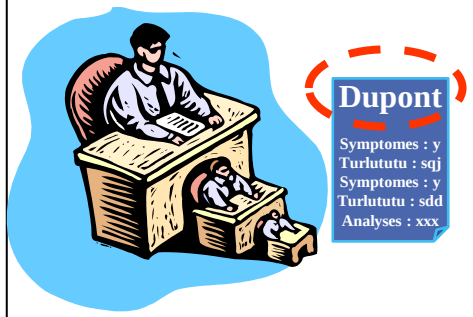
➔ plusieurs formats

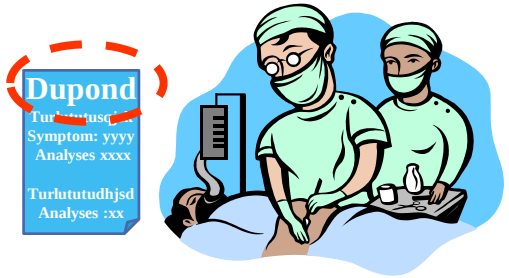
➔ plusieurs langages

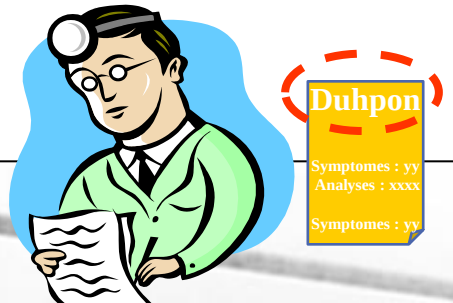
Problèmes

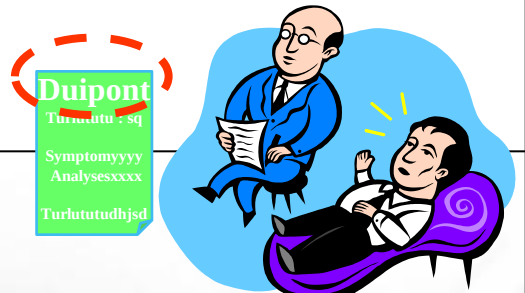
➔ **Difficultés de gestion**

Redondance (données)









Caractéristiques

Plusieurs applications

➔ plusieurs formats

➔ plusieurs langages

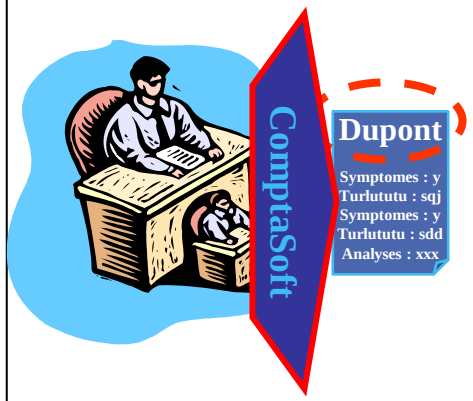
Redondance de données

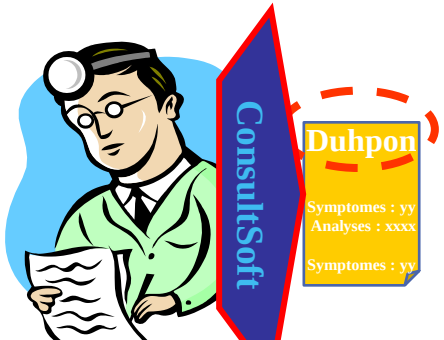
Problèmes

➔ **Difficultés de gestion**

➔ **Incohérence des données**

Interrogations





Caractéristiques

Plusieurs applications

➔ plusieurs formats

➔ plusieurs langages

Redondance de données

Pas de facilité d'interrogation

➔ Question ⇒développement

Problèmes

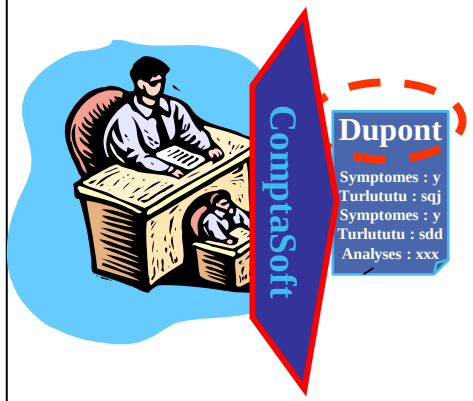
➔ **Difficultés de gestion**

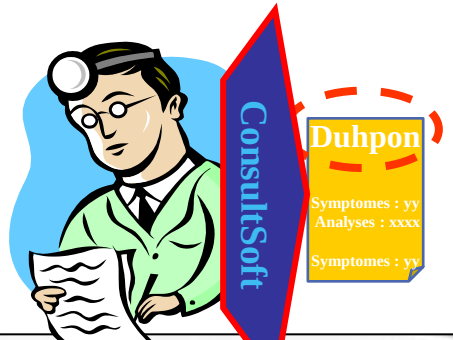
➔ **Incohérence des données**

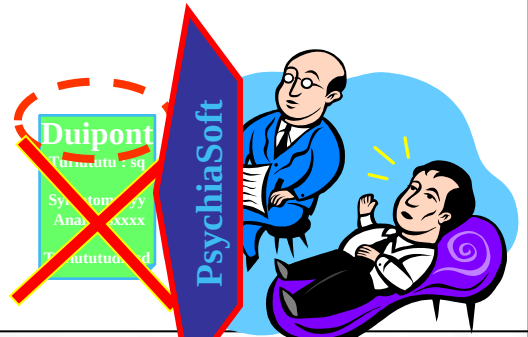
➔ **Coûts élevés**

➔ **Maintenance difficile**

Pannes ???







Caractéristiques

Plusieurs applications

➔ plusieurs formats

➔ plusieurs langages

Redondance de données

Pas de facilité d'interrogation

➔ Question ⇒développement

Redondance de code

Problèmes

➔ **Difficultés de gestion**

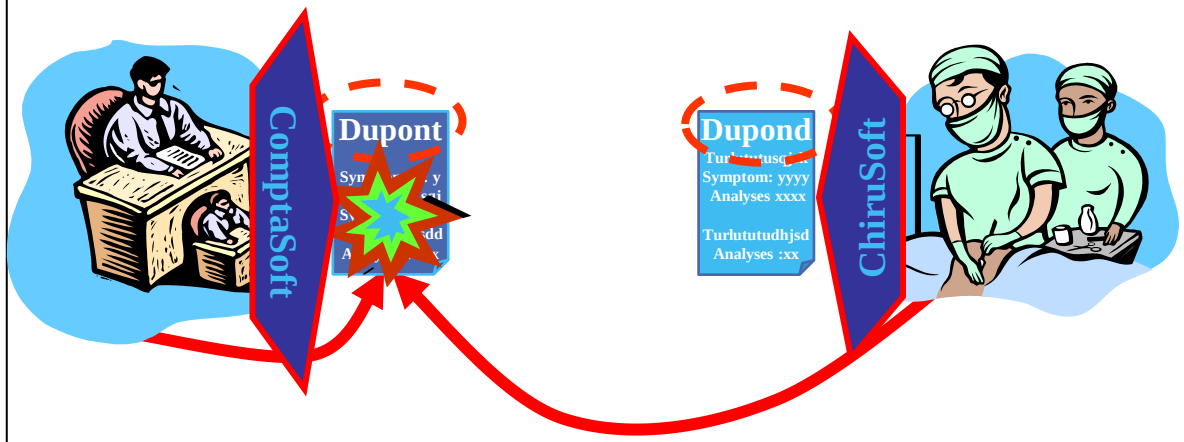
➔ **Incohérence des données**

➔ **Coûts élevés**

➔ **Maintenance difficile**

➔ **Gestion de pannes ???**

Partage de données



Caractéristiques

Plusieurs applications

➔ plusieurs formats

➔ plusieurs langages

Redondance de données

Pas de facilité d'interrogation

➔ Question ⇒développement

Redondance de code

Problèmes

➔ **Difficultés de gestion**

➔ **Incohérence des données**

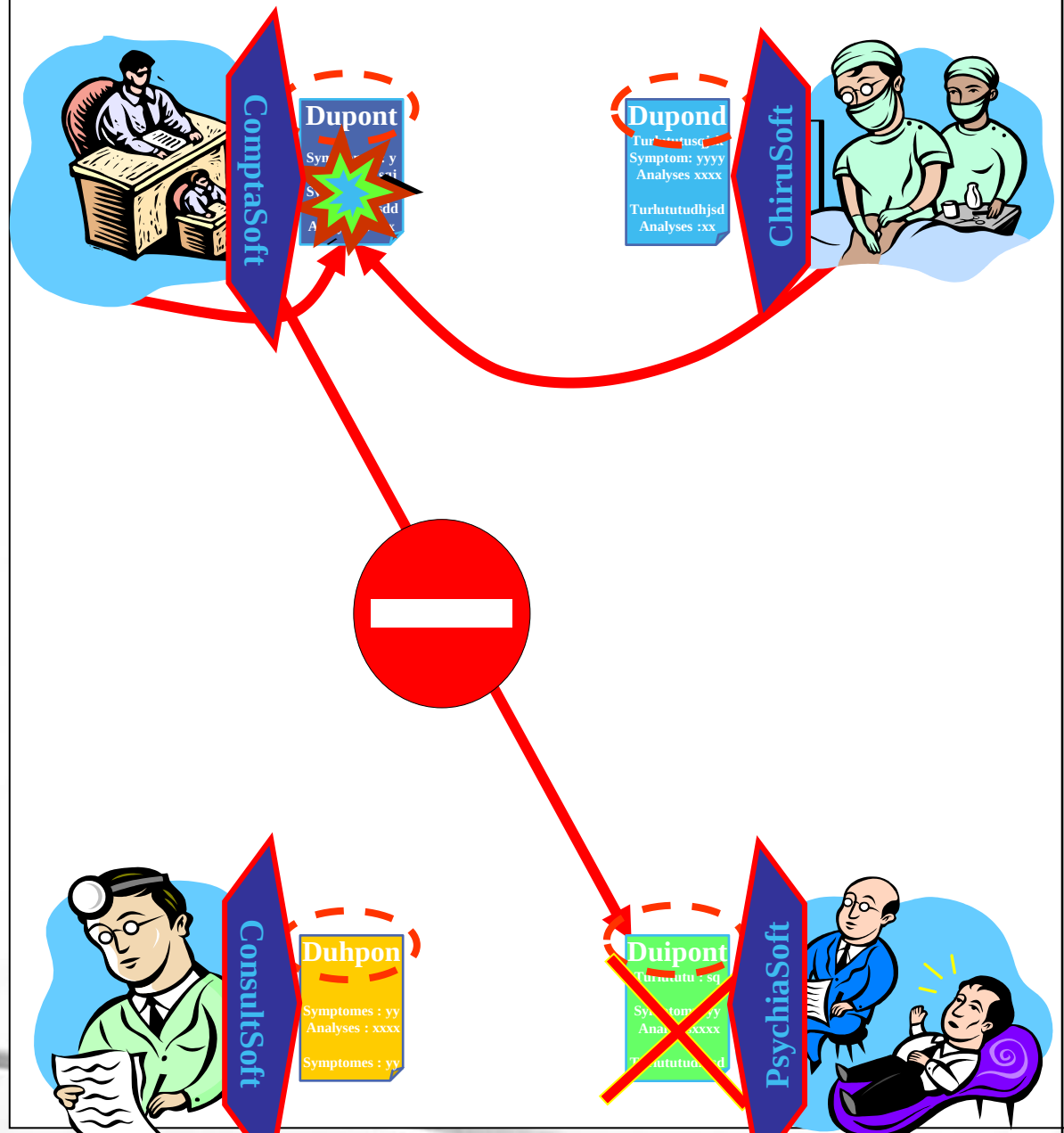
➔ **Coûts élevés**

➔ **Maintenance difficile**

➔ **Gestion de pannes ???**

➔ **Partage des données ???**

Confidentialité



Caractéristiques

Plusieurs applications

➔ plusieurs formats

➔ plusieurs langages

Redondance de données

Pas de facilité d'interrogation

➔ Question ⇒développement

Redondance de code

Problèmes

➔ **Difficultés de gestion**

➔ **Incohérence des données**

➔ **Coûts élevés**

➔ **Maintenance difficile**

➔ **Gestion de pannes ???**

➔ **Partage des données ???**

➔ **Confidentialité ???**

Les Bases de Données

Définitions

- Une base de données est un ensemble structuré de données :
  - Enregistrées sur des supports (Organisation et description de données)
  - Accessibles par l'ordinateur (Stockage sur disque)
  - Pour satisfaire simultanément plusieurs utilisateurs (Partage des données)
  - De manière sélective (Confidentialité)
  - En un temps opportun (Performance).
- Une base de données est une collection de données sur un domaine d'application particulier mémorisée par un ordinateur où les propriétés des données ainsi que les relations sémantiques entre ces données sont spécifiées.

Les Bases de Données

Base de données
Location des voitures

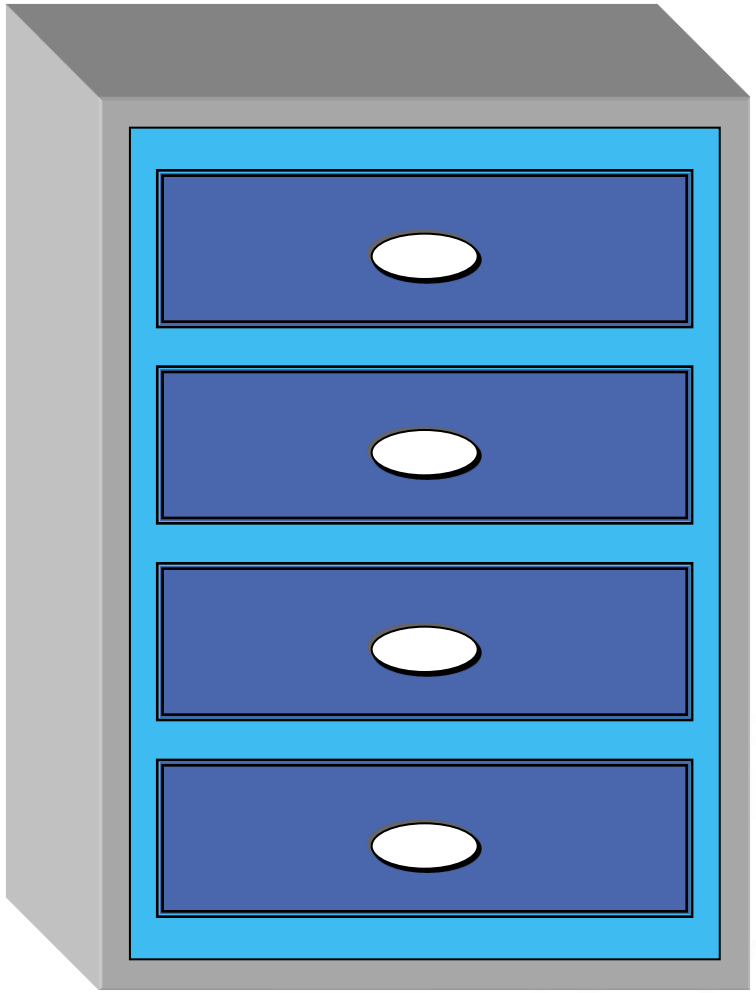


Table : Voiture

Table : Location

Table : Client

Table : Entretient

Les Bases de Données (Suite)

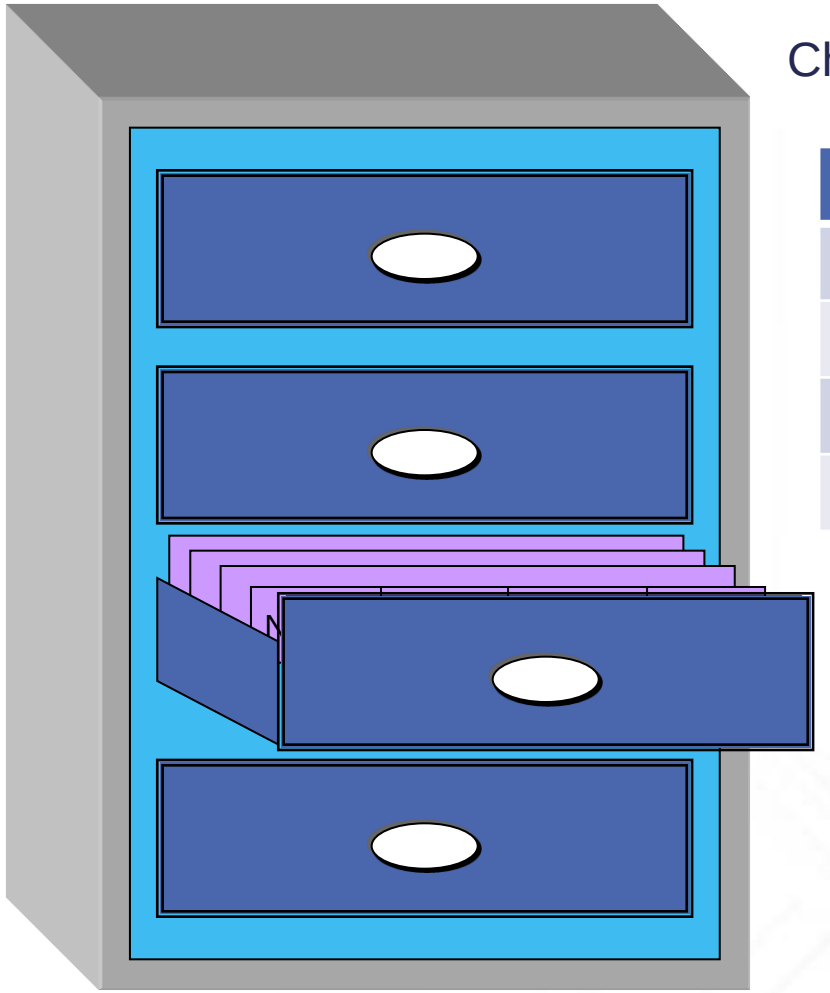
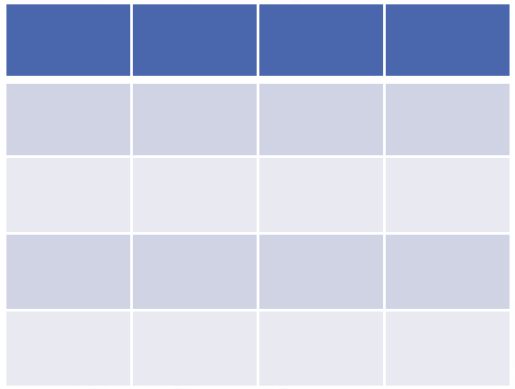


Table : Client

Champ



Enregistrement

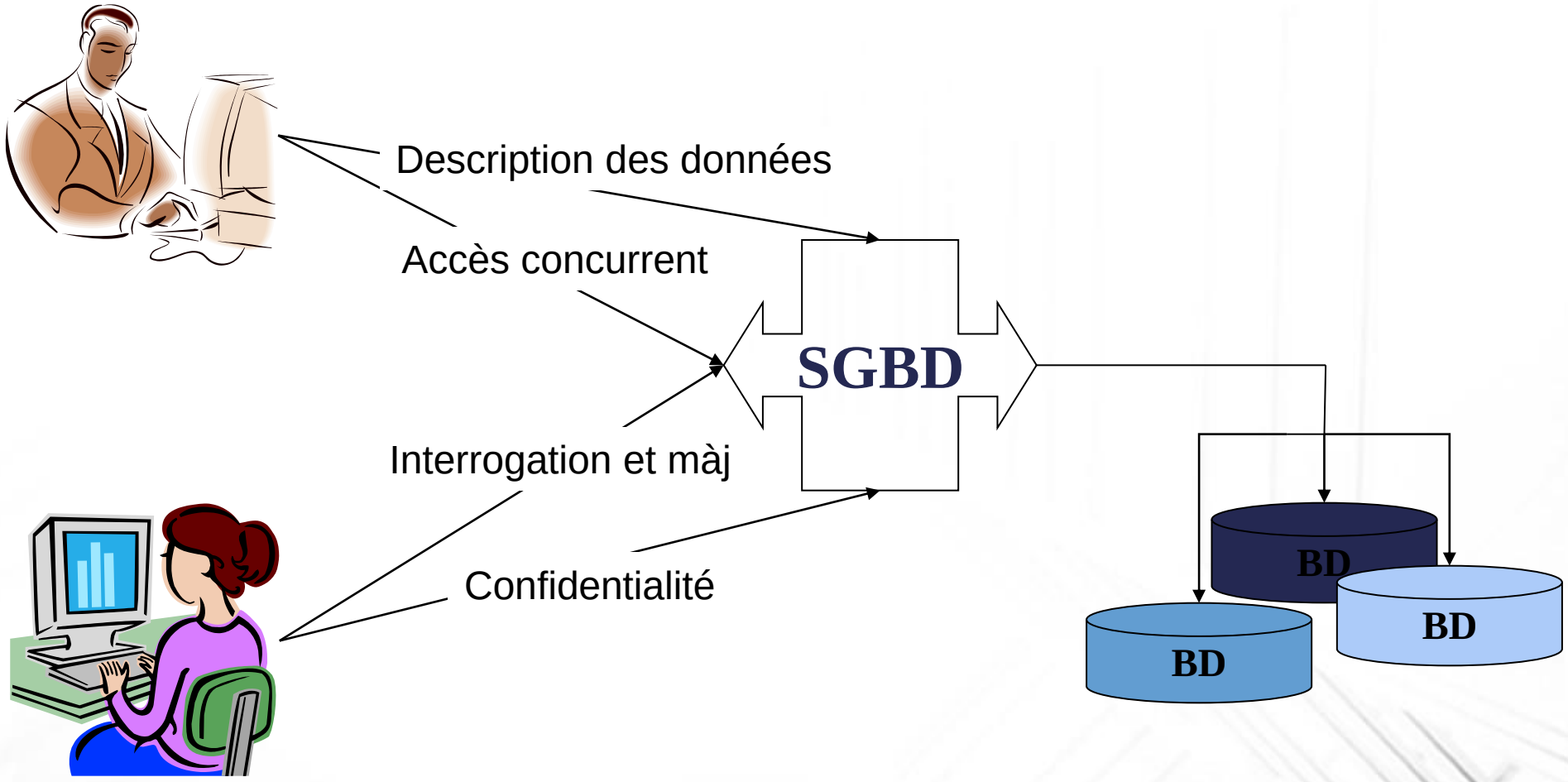
Les Bases de Données (Suite)

- Les B.D. doivent satisfaire les cinq critères suivants :
  - Bonne représentation du monde réel (une image aussi fidèle que possible de la réalité)
  - Non-redondance de l'information (une saisie unique de l'information)
  - Indépendance des programmes par rapport aux données (+rs applications partagent les mêmes données)
  - Sécurité et confidentialité des données (information ne pourrant être accessibles qu'aux personnes habilitées à en prendre connaissances) + (sécurité contre toute altération ou destruction)
  - Performances des applications

Les Bases de Données (Suite)

- Système de Gestion de Base de Données (SGBD):
  - Logiciel qui gère l'ensemble des fichiers constituant la B.D et qui permet à plusieurs utilisateurs de stocker et d'extraire les données de ces fichiers, dans des conditions d'intégrité et de confidentialité.
  - Prend en charge la structuration, le stockage, la mise à jour et la maintenance des données.
  - C'est, en fait, l'interface entre la base de données et les utilisateurs.

Les Bases de Données (Suite)



Les Bases de Données (Suite)

- Objectifs fondamentaux d'une BD
  - Centraliser l'information
    - Supprimer la redondance des données
    - Unifier l'opération de saisie des données
    - Centraliser les contrôles sur les données
  - Assurer l'indépendance données / traitements (D/T)
  - Permettre les liaisons entre +rs ensembles de données
    - Associer un joueur à son équipe,
    - Associer un fournisseur à l'ensemble de ses produits, …

Les Bases de Données (Suite)

- Objectifs fondamentaux d'un S.G.B.D
  - Indépendance physique
    - Permettre de réaliser l'indépendance des structures de stockage aux structures de données du monde réel.
  - Cohérence des données
  - Partage des données
  - Sécurité des données
  - Résistance aux pannes

Les Bases de Données (Suite)

- Exemple de SGBD
  - De nombreux SGBD sont aujourd'hui disponibles. La plupart sont dotés de capacités relationnelles.
  - On peut citer FoxPro, Access (Microsoft) et Paradox (Borland).
  - Oracle, Ingres, Informix et Sybase. DB2 (IBM).

Les Bases de Données (Suite)

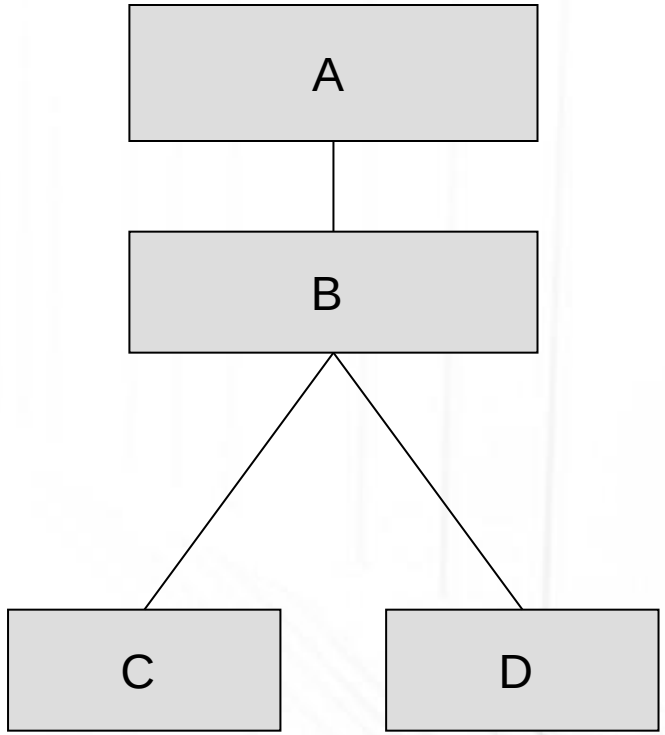
- Cycle de vie d'une B.D.
  - La conception
    - Définition des fonctionnalités
  - L'implantation
    - Réalisation effective de la base
  - L'exploitation
    - Utilisation et maintenance de la base.

Les modèles de données

- Les modèles des données les plus connus sont :
  - Le modèle hiérarchique,
  - Le modèle réseau,
  - Le modèle relationnel,
  - Le modèle objet.

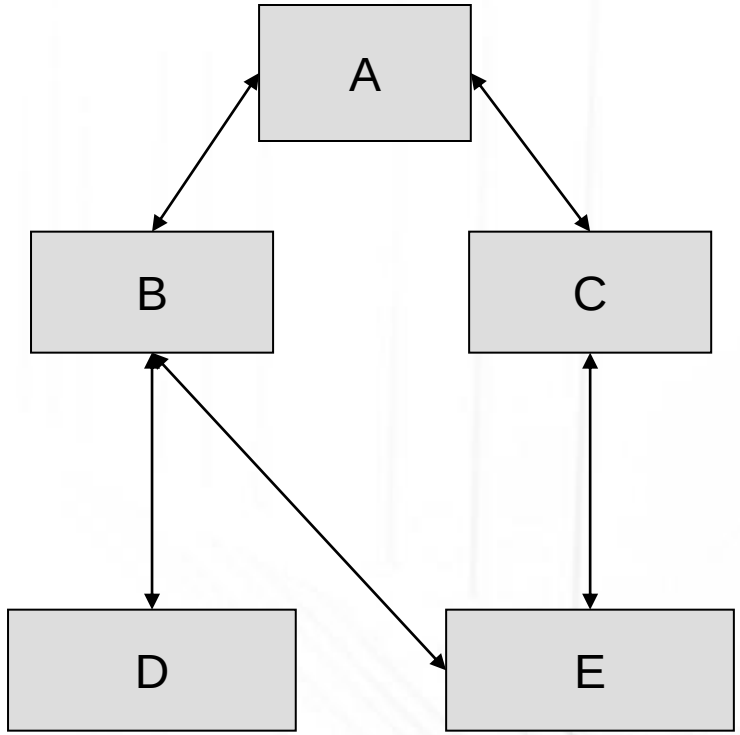
Les modèles de données

- Le modèle hiérarchique
  - Les données sont représentées sous forme d'une structure arborescente d'enregistrements.
  - Cette structure est conçue avec des pointeurs et détermine le chemin d'accès aux données.
  - Le seul type de lien possible est le lien père-fils.
  - Le premier nœud de l'arbre est appelé racine et les liens sont des branches.



Les modèles de données

- Le modèle réseau
  - La structure des données peut être visualisée sous la forme d'un graphe quelconque.
  - La structure de graphe utilise des pointeurs et détermine le chemin d'accès aux données.
  - Tous les types de lien sont possibles (1,1), (1,N), (N,1), (M,N).



Les modèles de données

- Pour le modèle hiérarchique et réseau, les programmes
  - Ne sont pas indépendants de la structure logique de la base et du chemin d'accès aux données.
  - Doivent indiquer le chemin d'accès aux données.
  - Utilisent un langage complexe pour travailler avec les données.
  - Si par exemple, on enlève un index, tous les programmes qui l'utilisent doivent être réécrits.

Les modèles de données

- Le modèle relationnel
  - Le modèle se présente comme un ensemble de relation d'où le nom modèle relationnel.
  - La théorie des SGBDR est fondée sur la théorie mathématique des relations.
  - Représentation très simple des données sous forme de tables constituées de lignes et de colonnes.
  - Plus de pointeurs qui figent la structure de la base.
  - Langage non procédural, puissant et simple d'emploi.
  - Langage SQL est un standard parmi ces langages.
  - Domine le marché des SGBD.

Les modèles de données

- Le modèle relationnel
  - Le modèle relationnel a montré ses limites pour le type de données complexes.
  - Les structures de données complexes sont éclatées par le modèle relationnel.
  - La reconstitution de la structure nécessite des opérations (jointures) lourdes et couteuses en performance.

Les modèles de données

- Le modèle objet
  - Le modèle objet permet de présenter les données sous forme d'objets. Les données sont enregistrées avec les procédures et les fonctions qui permettent de les manipuler.
  - Ils supportent la notion d'héritage entre classes d'objets.
  - Meilleures performances pour la gestion d'objets complexes (les pointeurs remplacent les jointures pour les structures hiérarchiques).

Plan du cours

- Chap. II : Le Modèle Relationnel
  - Les concepts de base
  - Normalisation d'une BD relationnelle
  - Processus de normalisation

Chapitre II. Le Modèle Relationnel

- Le modèle relationnel est le modèle le plus utilisé par les SGBDs actuels pour implémenter le niveau conceptuel.
- Les systèmes les plus répandus sont basés sur un modèle relationnel, Oracle et Ingres par exemple pour les gros systèmes, Access pour les micro-ordinateurs.
- Le modèle relationnel a été introduit par Codd en 1970.

Les concepts de base

- L'unité de base de représentation de l'information est la relation (encore appelée table)
- Une BD relationnelle se présente donc comme une collection de relations.
- Dans le modèle relationnel, chaque information doit appartenir à un domaine.

Les concepts de base

- Un domaine est un ensemble de valeurs caractérisé par un nom et qui correspond à un type élémentaire.
- C'est l'ensemble de valeurs que peut prendre un attribut.
- Par exemple, on y trouve les entiers, les réels, les chaînes de caractères, les dates, etc.

Les concepts de base

- Exemple :

  Cassette (code_cas : D1, titre_film : D2, type : D3, …)

  D1 : entier appartenant [1,200]

  D2 : chaîne de caractères de longueur < 20

  D3 : chaîne de caractères dans {policier, sciences fiction, action,…}

Les concepts de base

- le produit cartésien d'un ensemble de domaines D1, D2..., Dn est l'ensemble de tous les vecteurs <v1, v2..., vn> où, pour i variant de 1 à n, vi est une valeur de Di.

Les concepts de base

- Par exemple, le produit cartésien des domaines :
  DESIGNATION={ECROU, BOULON, VIS}
  COULEUR={BLANC, NOIR} et
  est composé des six vecteurs :

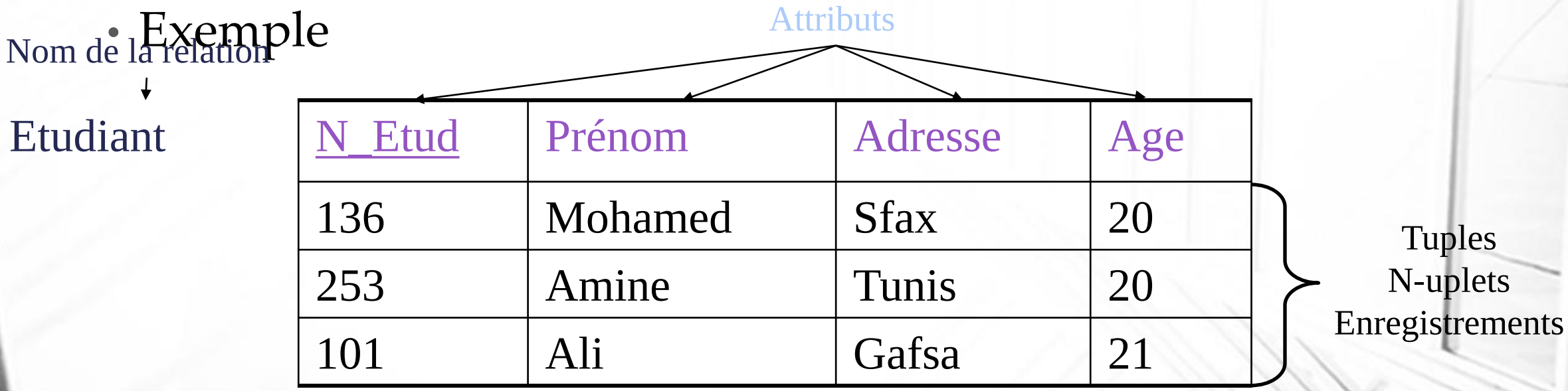
ECROU	BLANC
ECROU	NOIR
BOULON	BLANC
BOULON	NOIR
VIS	BLANC
VIS	NOIR

Les concepts de base

- Une Relation est un sous-ensemble du produit cartésien d'une liste de domaines caractérisé par un nom.
- Un Attribut est une colonne d'une relation caractérisée par un nom. L'attribut définit le rôle (signification) de la colonne.
- Les lignes d'une relation correspondent à des n-uplets de valeurs. Une ligne est aussi appelée tuple qui est un enregistrement dans une relation

Les concepts de base

- L'ensemble des n-uplets d'une relation est appelé extension de la relation (appelée aussi **instance** de de la relation).
- Exemple



Les concepts de base

- la Cardinalité d'une relation ***R*** est le nombre de tuples de ***R***.
- le Degré d'une relation ***R*** est le nombre d'attributs de ***R***.
- Exemple

Degré (Etudiant) = ?

Cardinalité (Etudiant) = ?

Les concepts de base

- le schéma d'une relation est le nom de la relation suivi de la liste des attributs et de la définition de leurs domaines.
- le schéma d'une relation représente son intention.
- Exemple :

Etudiant (N_Etud : Dnum, Nom : Dnom, Prénom : Dpré, Age : Dâge)

- La plupart du temps, lorsqu'on définit le schéma d'une relation, les domaines sont implicites et découlent du nom de l'attribut.
- Exemple : Etudiant(N_Etud, Nom, Prénom, Age)

Les concepts de base

- L'Identifiant d'une relation est un Attribut ou une combinaison d'attributs permettant de caractériser de manière unique chaque tuple de la relation.

Exemple : N_Etud est un identifiant pour la relation Etudiant.

- S'il existe plusieurs attributs ou combinaisons d'attributs possédant cette propriété d'identifiant, alors chacun d'eux est une clé candidate.

Les concepts de base

- La clé candidate choisie comme identifiant est appelée clé primaire. C'est l'identifiant (minimum) d'une relation.

  Exemple : N_Etud est un identifiant pour la relation Etudiant.

  - Par convention, la clé primaire d'une relation est soulignée dans un schéma de relation.
  - Exemple : Etudiant(<u>N_Etud</u>, Nom, Prénom, Age)

Les concepts de base

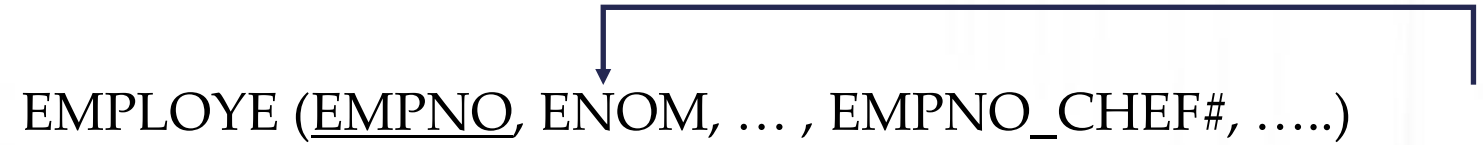
- Une clé étrangère est un ensemble d'une ou de plusieurs colonnes d'une relation qui fait référence à une clé primaire d'une (autre) relation.
  - Toutes les valeurs des clés étrangères apparaissent dans une (autre) relation comme valeurs d'une clé primaire.
  - Par convention, la clé étrangère d'une relation est suivie par le symbole # dans un schéma de relation.
  - Exemple

COMMANDES(NUMCDE,DATCDE,NUMCLI#,ETATCDE)

CLIENTS (NUMCLI, NOMCLI, ADRCLI)

Les concepts de base

- La clé étrangère et la clé primaire n'appartiennent pas nécessairement à deux relations distinctes.
- Exemple :



Les contraintes d'intégrité

- Contraintes d'intégrité
  - Un des avantages des bases de données par rapport à la gestion de fichiers traditionnelle réside dans la possibilité d'intégrer des contraintes que doivent vérifier les données à tout instant.
  - Ceci est possible grâce à la notion de **contraintes d'intégrité**.
  - Définition : Contraintes d'intégrité « sont des assertions qui doivent être vérifiées à tout moment par les données contenues dans la base de données »

Les contraintes d'intégrité

Trois types de C.I. obligatoires

- Contrainte de clé : une relation doit posséder une clé primaire
- Contrainte d'entité : un attribut d'une clé ne doit pas posséder de valeurs nulles (vides)
- Contrainte de référence (pour les clés étrangères)
  - C'est une contrainte exprimée entre deux tables.
  - Tout enregistrement d'une relation faisant référence à une autre relation doit se référer à un enregistrement qui existe.
  - Quand on désigne un attribut comme clé étrangère, les seules valeurs que peut prendre cet attribut sont celles qui sont déjà saisies dans la table qu'il référence.

Normalisation d'une BD relationnelle

- Un schéma de relation est décrit par la liste de ses attributs et de leurs contraintes d'intégrité.
- La constitution de la liste d'attributs du schéma ne peut pas se faire n'importe comment pour ne pas provoquer de redondance, avec toutes ses implications (perte de place, risques d'incohérence et de perte d'informations).
- Les formes normales des relations et les mécanismes pour les construire permettent d'obtenir des relations non redondantes.
- Les formes normales sont fondés sur les notions de clés de relations et de dépendances entre données.
- Les formes normales fournissent les conditions d'application d'un processus dit de normalisation qui mène à des formes « correctes » de relations qui sont les formes normales.

Normalisation d'une BD relationnelle

PRODUIT(Refproduit, LibelleProduit, PU, Quantité, NumService, Adresse, Capacité)

RefProduit	LibelleProduit	PU	Quantité	NumService	Adresse	Capacité
P1	CH7	23.510	300	S1	Sousse	9000
P1	CH7	23.510	500	S2	Tunis	6000
P3	VIS12	0.150	900	S4	Sousse	2000

- Cette relation présente certaines anomalies :
- Redondance : un produit apparaît autant de fois qu'il sera livré par un service
- Risques d'incohérences lors des mises à jour :

si l'on s'aperçoit que le libelle de P1 n'est pas CH7 mais CH1, il faudra veiller à mettre à jour tous les tuples contenant P1.

Il est nécessaire d'autoriser la présence de valeurs nulles dans une telle relation afin de pouvoir conserver dans la base des services sans produit.

si nous supprimons un produit qui soit le seul livré à un service donné, nous supprimons avec lui toute trace de ce service

Normalisation d'une BD relationnelle

- On peut dire qu'une Base de Données relationnelle est 'correcte' ou normalisée si :
  - Chaque relation décrit une information élémentaire avec les seuls attributs qui lui sont directement liés
  - Il n'y a pas des redondances d'informations qui peuvent produire des problèmes de mise à jour

- La relation Produit peut être décomposée en trois relations non redondantes
  - PRODUIT1 (RefProduit, LibelleProduit, PU)
  - PRODUIT21 (RefProduit#, NumService#, Quantité)
  - PRODUIT22 (NumService, Adresse, Capacité)

Normalisation d'une BD relationnelle

PRODUIT(RefProduit, LibelleProduit, PU, Quantité, NumService, Adresse, Capacité)

Décomposition 1

PRODUIT1(RefProduit, Libelle, PU)

P1	CH7	23.510
P3	VIS12	0.150

PRODUIT2 (RefProduit, NumService, Quantité, Adresse, Capacité)

Décomposition 2

PRODUIT22 (NumService, Adresse, Capacité)

S1	Sousse	9000
S2	Tunis	6000
S4	Sousse	2000

PRODUIT 21 (RefProduit#, NumService#, Quantité)

P1	S1	300
P1	S2	500
P3	S4	900

Normalisation d'une BD relationnelle

- Le résultat final de la décomposition est donc les relations suivantes :
- PRODUIT1 (RefProduit, LibelleProduit, PU)
- PRODUIT21 (RefProduit#, NumService#, Quantité)
- PRODUIT22 (NumService, Adresse, Capacité)

Dépendance fonctionnelle

- Définition
- Soit une relation R […, A, B, …], on dit qu'il existe une DF entre les 2 attributs A et B de la relation R, si à toute valeur de A il ne lui est associé qu'une seule valeur de B.
- On note une telle DF

  A → B (A détermine B ou B dépend fonctionnellement de A).

- Exemple
- PRODUIT (RefProduit, LibelleProduit, PU, Quantité, NumService, Adresse, Capacité)
- Pour cette relation, les dépendances fonctionnelles suivantes sont vérifiées :
  - RefProduit → LibelleProduit
  - NumService → Adresse, Capacité

Propriétés des dépendances fonctionnelles

- Des axiomes et des règles d'inférence permettent de découvrir de nouvelles dépendances à partir d'un ensemble initial. Dans ce que suit nous considérons R une relation. Les trois premières propriétés sont connues sous le nom « Axiomes d'Armstrong »
- Propriété 1 : Réflexivité

  $A \rightarrow A$ et si $\exists B \subseteq A$ alors $A \rightarrow B$

  Tout ensemble d'attributs détermine lui-même ou une partie de lui-même.
- Propriété 2 : Augmentation

  $A \rightarrow B$ alors $A,C \rightarrow B,C$

  Si A détermine B, les deux ensembles d'attributs peuvent être enrichis par un même troisième.
- Propriété 3 : Transitivité

Propriétés des dépendances fonctionnelles

- Propriété 4 : Union

  A → B et A → C alors A → B,C

- Propriété 5 : Pseudo-transitivité

  A→ B et C,B → D alors A,C → D

- Propriété 6 : Décomposition

  A→ B et C $\subseteq$ B alors A→ C

Dépendance fonctionnelle élémentaire

- Une Dépendance fonctionnelle A→ B est élémentaire si pour tout A'⊂A la dépendance fonctionnelle A'→ B n'est pas vraie.
- En d'autres termes, B ne dépend pas fonctionnellement d'une partie de A (A est la plus petite quantité d'information donnant B).
- Exemple :

produit pour déterminer le prix unitaire.

Dépendance fonctionnelle canonique

- Une dépendance fonctionnelle A→B est canonique si sa partie droite ne comporte qu'un seul attribut.
- Un ensemble F de dépendances fonctionnelles est canonique si chacune de ses dépendances est canonique.
- Exemple
  - NumService → Adresse, Capacité : n'est pas canonique

Clé d'une relation

- La clé d'une relation est l'ensemble d'attributs dont les valeurs permettent de caractériser les n-uplets de la relation de manière unique.
- Formellement :
- Un attribut ou une liste d'attributs A est une clé pour la relation R(A, B, C) si
  - B et C dépendent fonctionnellement de A dans R : A → B, C
  - et A → B, C est élémentaire.

- Une relation peut avoir plusieurs clés. Une clé sera choisie et désignée comme clé primaire. Les autres seront appelées clés candidates.

Processus de normalisation

- Les formes normales ont été définies pour permettre la décomposition des relations sans perte d'informations en utilisant la notion de dépendance fonctionnelle.
- La normalisation des relations est l'application d'un ensemble de règles prédéfinies, introduites dans le modèle relationnel afin de garantir la cohérence de la base lors des différentes opérations de manipulation et de MAJ.
- Le processus est basé sur 5 niveaux ou 5 formes normales.

Première Forme Normale (1FN)

- Une relation R est en 1ère FN si
  - Elle possède une clé
  - Tous ses attributs sont atomiques (non décomposables)
- Exemple

Cette relation peut par exemple être transformée vers la nouvelle relation :

- LIVRE (No-ISBN, Titre, Auteur1, Auteur2, Auteur 3, Editeur)

Première Forme Normale (1FN)

- ETUDIANT (Num, Nom, Prénom, Adresse(Rue, Ville))
- n'est pas en 1FN car l'attribut Adresse n'est pas atomique.
- Cette relation peut par exemple être transformée vers la nouvelle relation suivante :
- ETUDIANT (Num, Nom, Prénom, Rue, Ville)

Deuxième Forme Normale (2FN)

- Une relation R est en 2FN si et seulement si :
  - Elle est en 1FN,
  - Toutes les DF entre la clé et les autres attributs sont élémentaires, aucun attribut ne dépend d'une partie de la clé.
- Exemple
  - Soit la relation CLIENT avec ses DF

    CLIENT (NumCl, AdrCl, RefProduit, PU)

    F1 : NumCl, RefProduit → PU

    F2 : NumCl → AdrCl
  - La clé de la relation est (NumClt, RefProduit)
  - Suite à F2, une partie de la clé (NumClt) détermine un attribut n'appartenant pas à la clé. Cette relation n'est donc pas en 2FN. Elle pourra être décomposée en :

    CLIENT (NumCl, AdrCl)

    PRODUIT (RefProduit, NumCl #, PU)

Troisième Forme Normale (3FN)

- Une relation R est en troisième forme normale (3FN) si et seulement si :
  - Elle est en 2FN,
  - Tout attribut n'appartenant pas à une clé ne dépendra pas d'un attribut non clé : (Toutes les DF : sont directes pas de transitivité).
- La troisième forme normale permettra d'éliminer les redondances dues aux dépendances transitives.
- La décomposition en 3FN est sans perte d'informations et préserve les DF.

Troisième Forme Normale (3FN)

- Exemple1 :

  CLIENT (NumCl, ChiffreAffaire, Ville, Pays)

  Avec les dépendances fonctionnelles suivantes :

  - F1 : NumCl → ChiffreAffaire
  - F2 : NumCl → Ville
  - F3 : Ville → Pays

- La relation CLIENT n'est pas en 3FN à cause des dépendances fonctionnelles F3.
- Cette relation doit être décomposée en deux relations :
  - CLIENT (NumCl, ChiffreAffaire, Ville#)

Troisième Forme Normale (3FN)

- Exemple2 :

  VOITURE (NumVoiture, Marque, Type, Puissance, Couleur)

  - F1 : NumVoiture → Marque, Type, Puissance, Couleur
  - F2 : Type → Marque

- N'est pas en 3FN. En effet, l'attribut non clé TYPE détermine MARQUE (F2).
- Cette relation peut ainsi être décomposée en deux relations :
  - VOITURE (NumVoiture, Type#, Couleur, Puissance)
  - MODELE (Type, Marque)

Forme Normale de Boyce-codd (BCNF)

- Une relation R est en BCNF si et seulement si les seules dépendances fonctionnelles élémentaires qu'elle comporte sont celles dans lesquelles une clé détermine un attribut.
- En d'autres termes une relation est en BCNF, si :
  - Elle est en 3FN
  - Aucun attribut membre de la clé ne dépend fonctionnellement d'un attribut non membre de la clé.

Forme Normale de Boyce-codd (BCNF)

- Exemple :

  ADRESSE (Ville, Rue, CodePostal)

- Cette relation présente les DF suivantes :
  - Ville, Rue → CodePostal
  - CodePostal → Ville
- Elle est en 3FN.
- Cette relation n'est pas en BCNF car l'attribut ''Ville'' (qui fait partie de la clé) dépend fonctionnellement de CodePostal (qui est un attribut non membre de la clé).
- Décomposition :
  - ADRESSE (Ville, Rue)

Approche par synthèse

- Objectif
  - Obtenir un schéma directement en 3FN en partant d'un ensemble ***F*** de DF recensées
  - Automatiser le processus de normalisation

Approche par synthèse : Définitions

- **Attribut superflu (étranger)**
  Soit f : A1,…..Ai,…..An → Y ($f \in F$) L'attribut Ai est étranger dans *f* si on peut l'éliminer de la partie gauche de *f* càd si on peut générer la df f ': A1,…..Ai-1,Ai+1…..An → Y à partir de F et les propriétés des df.

- **DF redondante**
  Une df f $\in$ F est redondante si elle peut être dérivée à partir des autres dfs de F essentiellement par transitivité.

Approche par synthèse : étapes

1. Eliminer les attributs *superflus* (étrangers) des parties gauches des df
2. Eliminer les df *redondantes*
3. Regrouper les df qui ont une même partie gauche dans un groupe H_i
4. Fusionner 2 groupes H_i et H_j dans le cas où il existe des dépendances *mutuelles* $X \leftrightarrow Y$ *avec X partie gauche des dfs de Hi et Y partie gauche des dfs de Hj*
5. Construire une relation sur chaque groupe (Précisez clé primaire et étrangère)

Approche par synthèse : exemple

1- Eliminer les attributs superflus

f6 contient un attribut superflu et sera simplifiée :

f1 : A → B

f6 : A, B, E → F + pseudo-transitivité ⇒ f6': A, E → F

2- Eliminer les df redondantes

f1 et f3 génère f2 par transitivité

f1: A → B et f3: B → C ⇒ f2: A → C

3- Regrouper les df selon les parties gauche

H1 : { A → B }
H2 : { B → C ; B → D }
H3 : { D → B }
H4 : { A, E → F }

$\mathcal{F}$ = { f1 A → B
~~f2 A → C~~
f3 B → C
f4 B → D
f5 D → B
f6' A,E → F }

Approche par synthèse

4- Fusion des groupes ayant des df mutuelles

B et D se déterminent mutuellement

H2 U H3 : { B → C ; B → D ; D → B }

5- Construction des relations

R1 (<u>A</u>, B#)

R2 (<u>B</u>, C, D)

R3 (<u>A#, E</u>, F)

On obtient un schéma en 3FN.

Plan du cours

- Chap. III : Algèbre Relationnelle
  - Qu'est-ce que l'algèbre relationnelle ?
  - Principales opérations
  - Arbres d'expression pour l'algèbre relationnelle

Qu'est-ce que l'algèbre relationnelle ?

- Une collections d'opérations, chacune agissant sur une ou deux relations et produisant une relation en résultat.
- Un langage pour combiner ces opérations.
- L'algèbre relationnelle est à l'origine de SQL.

Principales opérations

- Opérations ensemblistes (binaires)
  - $\cup$: union
  - $\cap$: intersection
  - X : produit cartésien
  - – : différence
- Opérations de base de données
  - π : projection
  - σ : sélection
  - ∞ : jointure
  - ϱ : renommage

Union (∪)

- Définition
  - Définition : opération entre deux relations de même schéma qui retourne une relation de même schéma contenant l'ensemble des tuples qui appartiennent à au moins une des deux relations.
  - Remarque : pas de duplication de tuple

Union (∪)

- Description :
  - Type opération: binaire
  - Syntaxe : R ∪ S
  - Notation fonctionnelle : Union (R,S) ou OR (R, S)
  - Sémantique : réunit dans une même relation les tuples de R et ceux de S (sans doublons)
  - Schéma : schéma(R ∪ S) = schéma(R) = schéma(S)
  - Pré-condition : schéma(R) = schéma(S)

Union (∪) : exemple

Supposons que nous disposons de 2 tables produit : produit1 et produit2 exprimant le fait que les produits sont stockés dans deux dépôts différents.

Question : Lister tous les produits.

Réponse : Réaliser l'union des deux tables de produit.

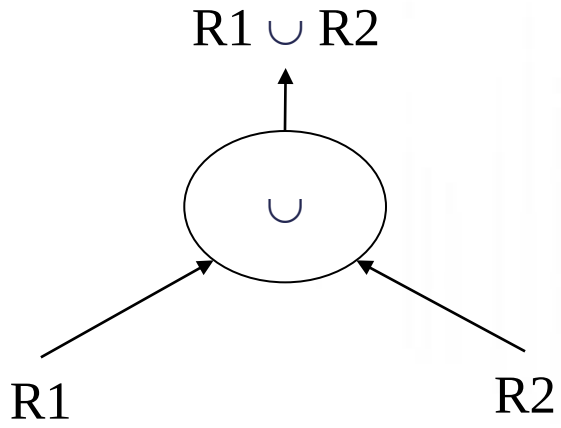
Produit 1

NP	**LibP**	**Coul**
P001	Robinet	Gris
P002	Prise	Blanc

Produit 2

NP	**LibP**	**Coul**
P003	Câble	Blanc
P004	Peinture	Blanc

Union (∪) : Représentation graphique



Intersection (∩)

- Définition
  - Définition : opération entre deux relations de même schéma qui retourne une relation de même schéma contenant l'ensemble des n-uplets qui appartiennent à la fois aux deux relations.

Intersection (∩)

- Description :
  - Type opération: binaire
  - Syntaxe : R ∩ S
  - Notation fonctionnelle : Inter (R,S) ou INTERSECT (R, S) ou AND (R,S)
  - Sémantique : sélectionne les tuples qui sont à la fois dans R et S
  - Schéma : schéma (R ∩ S) = schéma (R) = schéma (S)
  - Pré-condition : schéma(R) = schéma(S)

Intersection (∩) : exemple

Supposons que nous disposons de 2 tables produit produit1 et produit2 donnant respectivement les produits achetés par le client1 et le client2.

Question : Lister tous les produits identiques achetés par les 2 clients.

Réponse : Réaliser l'intersection des deux tables produit1 et produit2.

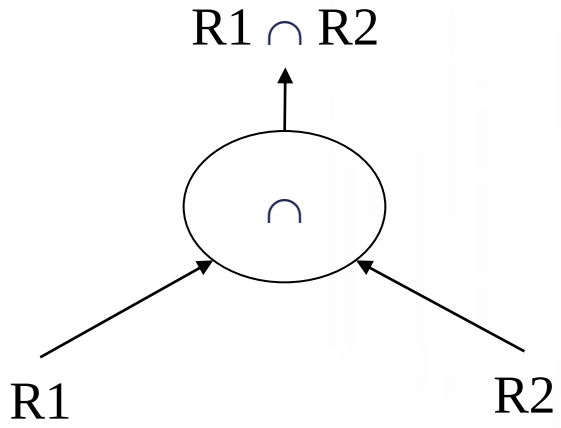
Produit 1

NP	LibP	Coul
P001	Robinet	Gris
P002	Prise	Blanc

Produit 2

NP	LibP	Coul
P002	Prise	Blanc
P004	Peinture	Blanc

Intersection (∩) : Représentation graphique



Différence (–)

- Définition
  - Définition : opération entre deux relations de même schéma qui retourne une relation de même schéma contenant l'ensemble des tuples appartenant à la première et n'appartenant pas à la deuxième
  - Remarque : opération non commutative

Différence (–)

- Description :
  - Type opération: binaire
  - Syntaxe : R - S
  - Notation fonctionnelle : Diff (R,S), MINUS(R,S)
  - Sémantique : sélectionne les tuples de R qui ne sont pas dans S
  - Schéma : schéma (R - S) = schéma (R) = schéma (S)
  - Pré-condition : schéma (R) = schéma (S)

Différence (–) : exemple

Supposons que nous disposons de 2 tables produit : produit1 et produit2 donnant respectivement les produits achetés par le client1 et le client2

Question : Lister tous les produits achetés par le client1 et non achetés par le client 2.

Réponse : Réaliser la différence entre les deux tables de produit.

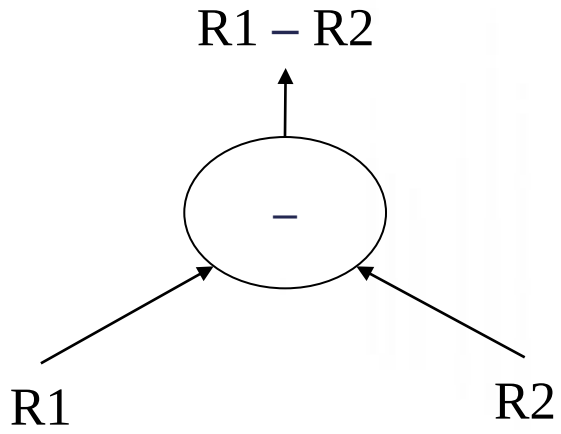
Produit 1

NP	LibP	Coul
P001	Robinet	Gris
P002	Prise	Blanc

Produit 2

NP	LibP	Coul
P002	Prise	Blanc
P004	Peinture	Blanc

Différence (–) : Représentation graphique



Produit cartésien (X)

- Définition
  - Définition : opération entre deux relations n'ayant pas d'attributs de même nom, qui retourne une relation ayant pour schéma la concaténation des deux schémas et contenant toutes les concaténations possibles des tuples des deux relations
  - Remarque : opération commutative

Produit cartésien (X)

- Description :
  - Type opération: binaire
  - Syntaxe : R X S
  - Notation fonctionnelle : PROD(R,S) ou PRODUCT(R,S)
  - Sémantique : chaque tuple de R est combiné avec chaque tuple de S
  - Schéma : schéma (R X S) = schéma(R) ∪ schéma(S)
  - Pré-condition : R et S n'ont pas d'attributs de même nom (sinon, renommage des attributs avant de faire le produit).

Produit cartésien (X) : exemple

Supposons que nous disposons de 2 tables : produit et client.
Question : Lister tous les achats possibles des clients (produits pouvant être achetés par tous les clients).
Réponse : Réaliser le produit cartésien entre les deux tables produit et client.
Pour simplifier, nous avons réduit le nombre de tuples.

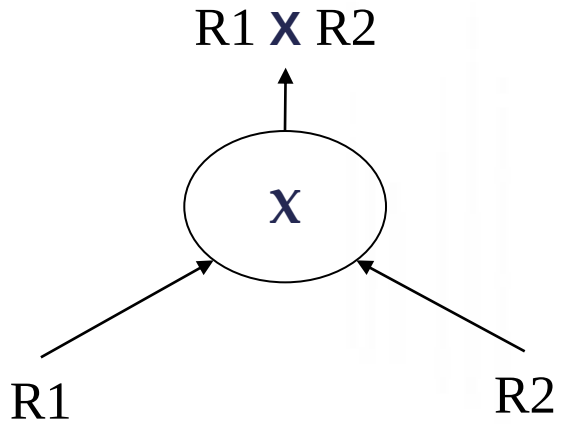
Produit

NP	LibP
P001	Robinet
P002	Prise

Client

NCl	NomCl
CL01	Batam
CL02	AMS

Produit cartésien (X) Représentation graphique



Renommage ρ

- Définition
  - Définition : Le renommage permet de renommer les attributs d'une relation pour résoudre des problèmes de compatibilité entre noms d'attributs de deux relations opérandes d'une opération binaire.

Renommage ρ

- Description :
  - Type opération: unaire
  - Syntaxe : ϱ [ancien_nom : nouveau_nom] R
  - Sémantique : les tuples de R avec un nouveau nom de l'attribut
  - Schéma : schéma (ϱ [n, m] R) le même schéma que R avec n renommé en m
  - Pré-condition : le nouveau nom n'existe pas déjà dans R

Situation actuelle

- Résumé de la situation actuelle.
- Utilisez une liste rapide, discutez oralement des détails.

Origines de cette situation

- Faits historiques pertinents.
- Hypothèses d'origine qui ne sont plus valables.

Options disponibles

- Présentez les stratégies alternatives.
- Dressez une liste des avantages et inconvénients de chacune.
- Indiquez le coût de chaque option.

Recommandations

- Recommandez une ou plusieurs stratégies.
- Donnez une synthèse des résultats si les propositions sont respectées.
- Indiquez les étapes suivantes.
- Identifiez les éléments d’action.