# GUIDE
# PRATIQUE

# MODÈLE CONCEPTUEL
# DES DONNÉES

# MODÈLE LOGIQUE
# DES DONNÉES STANDARD

# MODÈLE LOGIQUE
# DES DONNÉES OPTIMISÉ

# Présentation théorique de Merise

## Objectifs

- Définir, analyser, concevoir et spécifier tout projet d'organisation d'un système d'information
- Ni méthode de conduite de projet, ni méthode de programmation ou d'algorithmique
- En aval du schéma directeur, en amont de la réalisation

## Principes

- Approche globale, intégrant tous les sous-systèmes
- Conception descendante, partant des finalités de chaque activité
- Etude indépendante des données et des traitements, puis rapprochement pour valider l'étude des données avec les résultats de l'étude des traitements, et réciproquement.
- Approche par étapes (Conceptuelle, puis logique, enfin opérationnelle)
- Recherche des invariants du système d'informations
- Utilisation d'un formalisme facilitant la lecture et la communication



A partir des deux principes de séparation de l'analyse des données et de l'analyse des traitements d'une part, et d'une démarche en trois étapes, on obtient les questions à se poser dans le tableau suivant :

| | Analyse des données | Analyse des traitements |
|---|---|---|
| Niveau conceptuel | Quelles informations manipule-t-on ? | Que veut-on faire ? |
| Niveau logique | Comment structurer ces données ? | Qui fait quoi, où, quand ? |
| Niveau physique | Où les stocker ? | Comment ? |

A chacune de ces six questions, il s'agira d'amener des réponses. Le tableau suivant présente les documents qu'e la méthode Merise produit pour y répondre.

| | Analyse des données | Analyse des traitements |
|---|---|---|
| Niveau conceptuel | Modèle conceptuel des données (M. C. D.) | Modèle conceptuel des traitements (M. C. T.) |
| Niveau logique | Modèle logique des données (M. L. D.) | Modèle organisationnel des traitements (M.O.T.) |
| Niveau physique | Tables et index | Procédures |

Dans le cadre de l'utilisation d'un S.G.B.D., le concepteur est déchargé de l'implantation physique des tables. D'autre part, Merise ne guide pas le concepteur dans la production des procédures, car elles sont dépendantes du choix du système, des outils et des machines. Les seuls niveaux analysés sont donc les niveaux conceptuel et logique.
*L'expérience m'a amené à douter de l'efficacité de l'analyse des traitements (M.C.T et M.O.T)). De plus cette conception est en partie remise en cause par les technologies objet développées dans les outils modernes. Ce cours ne fera donc qu'effleurer ces chapitres, et se concentrera sur les M.C.D. et M.L.D. Enfin, les optimisations et ajustements nécessaires en fin d'analyse seront étudiés. Nous passerons ainsi d'un M.L.D "pur" à un M.L.D. optimisé (on pourrait presque dire "un M.L.D. perverti".*

# Guide pratique de Merise

*N.B : ce document est un support de cours, dont le but est d'aider à structurer et à mémoriser la démarche présentée pendant le cours. Son exploitation en tant que document autonome risque d'amener à bien des incompréhensions. Certaines notions (comme la définition précise des termes o"bjet du MCD", "Nombre d'occurences d'un attribut", "acteur" ou même "Donnée" ) ne sont donc pas définies dans ce document. De même, ce document est limité à un seul exemple, présenté pendant le cours, car l'interactivité entre enseignant et stagiaires paraît indispensable à la compréhension de s termes employés et à la mise en situation des exemples.*

# I - La réalisation d'un M.C.D.

## I.1 - Ce qu'on attend d'un M.C.D.

But :

- Il s'agit de représenter, par un formalisme précis et en grande partie standardisé, l'ensemble des informations que l'on doit traiter pour répondre aux attentes du projet défini en amont de l'analyse (dans le schéma directeur).

principes :

- IL FAUT OUBLIER LES MOYENS QUI SERONT MIS EN ŒUVRE POUR LA RÉALISATION. (il s'agit uniquement de décrire le problème à traiter, et pas du tout de préciser, simplifier ou guider les choix qu'on sera plus tard amenés à faire)
- Par "moyens mis en œuvre", il faut entendre machines et systèmes d'exploitation, mais aussi S.G.B.D, langages, outils et aussi culture informatique et maîtrise des produits par les développeurs. Tous ces points doivent impérativement être oubliés dans cette phase.
- Chacune des huit étapes décrites répond à une question élémentaire. Il ne faut surtout pas essayer de préparer le terrain pour les étapes suivantes. Il faut modestement se concentrer sur la seule question trairtée par cette étape.

Remarques :

# I.2 - Les huit étapes de la réalisation d'un M.C.D.

## I.2.A - Le dictionnaire des données (abr : DDD)

But :
- collecter l'ensemble des données (ou attributs) manipulées par le système.

Moyens :
- collecter au moins deux occurence de chacun des documents, écrits ou non, manipulés par chacun des acteurs.
- Pour chaque type de document, analyser l'ensemble de ses occurences, en repérant chaque zone dont le contenu peut être amené à varier d'une occurence de ce document à l'autre.
- Pour chaque type de document, analyser l'ensemble de ses occurences, en repérant chaque zone dont le contenu peut être amené à varier d'une occurence de ce document à l'autre, et qui doit être mémorisé. Une telle zone sera qualifiée de donnée. Donner un nom à chacune de ces zones, et l'ajouter dans le dictionnaire des données. Chacun de ces noms sera qualifié d' "attribut".

précautions :
- Il s'agit uniquement de collecter des données. Le seul impératif est de ne pas oublier une donnée. Tout classement, toute simplification, toute optimisation est à proscrire.
- Comme dans chacune des autres étapes, ne jamais se demander "comment va-t-on faire ?", mais décrire uniquement "ce qu'il y a".

Limites :
- Ce n'est qu'à la validation du M.C.D. à l'aide du M.C.T. qu'on peut être sûrs d'avoir effectivement obtenu le M.C.D. correspondant au problème. Autrement dit, cette première étape du M.C.D. ne peut être automatisée.
- On ne connaît pas de moyens de définir d'une façon non ambigüe "l'univers du discours", c'est-à-dire les données qui font ou ne font pas partie du problème à traiter.

Remarques :

EXEMPLE : Facturation classique

Dictionnaire des données :

Nom du client
N° de référence du produit
Prénom du client
Date facture
quantité achetée
N° de facture
Total facture
Prix unitaire HT
Montant TVA
Prix unitaire TTC
Adresse du client
Total article
Nombre d'articles
Mode règlement

## I.2.B - Epuration du dictionnaire des données

But :
- Epurer l'ensemble des attributs obtenus à l'étape précédente, en vérifiant que chaque donnée correspond à un "atome" d'informations, indivisible et indépendant des autres données.
  IL NE FAUT AUCUNE REDONDANCE (DIRECTE OU INDIRECTE) DANS UN DDD ÉPURÉ.

Moyens :
- Passer en revue, l'un après l'autre et sans ordre préétabli, chacun des attributs du DDD et vérifier les points suivants :
- Vérifier les synonymes : la même donnée utilisée sous deux termes différents par deux acteurs différents.
- Vérifier les homonymes : deux données différentes utilisées sous le même terme par deux acteurs différents.
- Vérifier les dépendances directes : une donnée qui peut être obtenue à partir d'autres données (exemple : prix unitaire HT, TVA, prix unitaire TTC : une de ces données doit être épurée).
- Vérifier les dépendances indirectes et les données calculées : une donnée obtenue comme totalisation ou comptage d'autres données (exemple : nombre de factures, ou chiffre d'affaires d'un client).
- Epurer les paramètres qui ne sont pas des données (exemple : la date de début et la date de fin d'un état récapitulatif sur une période).

précautions :
- La principale difficulté concerne les difficultés de communication. Est-on bien sûr que ce qu'on a entendu a le même sens pour nous et pour notre interlocuteur ?
- Se méfier des données du genre "Nombre de " ou "Totalisation" : elles peuvent presque toujours être calculées à partir d'autres données plus élémentaires, et doivent de ce fait être épurées.

Limites :
- Fondamentalement, les mêmes que pour l'étape 1.2.A.
- On est obligés de s'en remettre à son bon sens pour déterminer si oui ou non une donnée est un atome d'information indivisible dans l'univers du discours considéré.

Remarques :

EXEMPLE : Facturation classique

Dictionnaire des données épuré :

Nom du client
N° de référence du produit
Prénom du client
~~Date facture~~ -> JourMois Facture
Année Facture
quantité achetée
~~N° de facture~~ -> ~~Année Facture~~
N° Séquentiel
~~Total facture~~
Prix unitaire HT
Désignation
~~Montant TVA~~ ~~Taux TVA~~
~~Prix unitaire TTC~~
~~Adresse du client~~ -> 1° ligne adresse
2° ligne adresse
CP
Ville
~~Total article~~
~~Nombre d'articles~~
Mode règlement

## I.2.C - Mise en évidence des objets

But :
- Découper l'ensemble des attributs du dictionnaire des données épuré en différentes unités logiques, ici appelées "Objets du M.C.D".

Moyens :
- Reporter dans l'ébauche du M.C.D, l'un après l'autre et sans ordre préétabli, chacun des attributs du DDD, en l'attachant si possible avec l'un des objets déjà reconnus. Si cela n'est pas possible, créer un nouvel objet composé (pour l'instant) de ce seul attribut.
- Pour savoir si un attribut peut être attaché un objet, décrire dans un gand tableau l'ensemble des occurences des différents attributs correspondant à l'état du système à un instant t1, à un instant t2…
  Vérifier si le nombre d'occurences de l'attribut en cours de traitement est, pour chacun de ces instants, le même que le nombre d'occurences du premier objet déjà reconnu. Si ce n'est pas le cas, se poser la même question pour le même attribut et l'objet suivant.

précautions:
- Faire preuve de modestie et SE FIER À LA TECHNIQUE PLUTÔT QU'AU BON SENS : l'expérience montre que celui-ci est souvent mis endéfaut au cours de cette étape, et qu'il peut souvent amener à regrouper dans des objets des attributs ayant un nombre d'occurences plus élevé que celui de l'objet, et parfois à dissocier des attributs qui ont tous le même nombre d'occurences.
- Une erreur courante consiste à choisir l'attribut qu' on analysera au lieu de prendre le premier qui vient : ce choix "innocent" de l'attribut à traiter consiste en fait à considérer le problème du regroupement de cet attribut comme déjà réglé avant même de l'attaquer.
- En fonction de l'univers du discours, une donnée prenant un nombre fini et prédéterminé de valeurs distinctes pourra ne pas être considérée comme un objet mono-attribut *(cf exemple : Mode règlement)*.
- Chaque attribut doit figurer une fois et une seule dans le M.C.D.
- Une difficulté particulière : deux occurences d'un attribut ayant la même valeur. S'agit-il d'une seule occurence utilisée 2 fois ou de deux occurences distinctes ? A un niveau théorique, on pourrait toujours affirmer que deux valeurs semblables concernent toujours une seule occurence. Mais c'est souvent improductif. Il faut amener cette téponse dans le cadre de notre univers du discours, et reformuler la question de la manière suivante : si à deux occurences d'un attribut A correspond la même valeur pour l'attribut B, est-ce que, de ce fait, à ces deux occurences de l'attribut A correspondra la même valeur d'un autre attribut C ? Si oui, on a affaire à une seule occurence de l'attribut B, et le nombre d'occurences de A et B n'est donc pas le même *(cf exemple : à 2 "n° séquentiel" distincts correspond la même année de facturation. Mais ceci n'impose aucune valeur commune aux autres attributs du dictionnaire. On peut donc considérer qu'on a deux occurences distinctes de "Année de facturation" qui se trouvent avopir la même valeur, et donc que le nombre d'occurences de "n° séquentiel" est le même que le nombre d'occurences de Année de facturation"*

Limites :
- pas de limites : A partir de cette étape commence une démarche rigoureuse, qui nous permettra de systématiser l'analyse. A partir d'un dictionnaire des données épuré exact, tout merisien pourra obtenir un M.C.D, puis un M.L.D juste.

Remarques :

EXEMPLE : Facturation classique

Ebauche du M.C.D :

Nom
Prénom
1° ligne adresse
2° ligne adresse
CP
Ville

Réf. produit
Désignation
Prix unitaire HT

Qantité

Mode règlement

JourMois facture
Année facture
N° séquentiel

## I.2.D - Reconnaître ET IDENTIFIER les entités

But :

- Reconnaître ceux qui, parmi les objets obtenus à l'étape précédente, peuvent être "identifiés en interne", c'est-à-dire tous les objets dont les occurences pourront être repérées sans ambiguïté par le simple examen des occurences de leurs attributs.

Moyens :

- Pour qu'un objet soit identifiable, il faut et il suffit qu' il ne puisse pas y avoir deux occurences de cet objet pour lesquelles tous les attributs auront les mêmes valeurs (on pourra donc reconnaître chaque occurence en examinant les valeurs portées par ses attributs puisque l'ensemble de ces valeurs est distinct d'une occurence à l'autre).
- Un objet identifiable à partir de ses attributs est appelé une entité. Choisir un nom pour cette entité. L'entourer d'un rectangle surmonté du nom choisi.
- Pour cette entité, il faudra ensuite déterminer un sous-ensemble le plus limité possible de ses attributs (sous-ensemble souvent, mais non nécessairement limité à un seul attribut), sur lequel deux occurences de l'objet ne pourront avoir des valeurs distinctes (voir exemples). La concaténation de cas attributs pourra donc servir à identifier chaque occurence de l'entité. Cette concaténation sera appelée "Identifiant". Le ou les attributs omposant cet identifiant seront soulignés et placés en début d'entité.

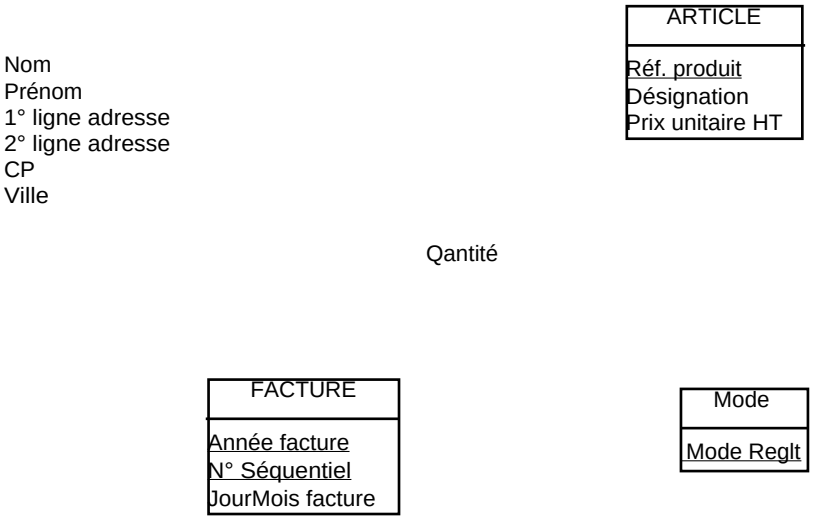
précautions :

- Ici aussi, SE FIER À LA TECHNIQUE PLUTÔT QU'AU BON SENS : il arrive qu'un choix d'identifiant paraisse évident et soit erronné.

Remarques :

EXEMPLE :

Ebauche du M.C.D :



## I.2.E - identifier les autres objets

But :

- Reconnaître les dépendances entre objets qui permettront d'identifier les objets qui ne peuvent l'être par eux-mêmes.

Moyens :

- L'étape précédente a permis de définir un identifiant pour chaque entité. Parmi les objets non identifiés, il faudra reconnaître ceux qui seraient identifiables si on essayait, à l'aide d'une redondance, de leur adjoindre un ou des identifiants déjà reconnus.
- Trois cas de figure se présentent :
  - L'objet est identifiable par deux ou plusieurs identifiants externes : l'objet est alors une relation porteuse de données. On reconnaît cette propriété de la manière suivante : si l'on dupliquait ces identifiants externes dans l'objet considéré, il y aurait, pour chaque occurence de l'objet, une seule occurence de chaque identifiant externe, et l'ensemble de ces identifiants externes est discriminant i.e : il n'y a pas deux occurences de l'objet pour lesquelles l'ensemble de ces identifiants prend les mêmes valeurs).
    Pour nommer cette relation, on choisit un verbe correspondant à l'action qui lie les entités ayant fourni les identifiants de la relation. On surmonte la relation de son nom, puis on l'entoure d'un cercle, et on relie cette relation à chacune des entités ayant fourni leur identifiant.
  - L'objet est identifiable par la combinaison d'un identifiant externe, et un ou plusieurs attributs internes : l'objet est alors une entité relative (à la fois une entité et une relation), qu'on nomme, et qu'on entoure d'un rectangle en pointillés, relié à l'entité identifiante par une flèche partant de la sous-entité.
    La reconnaissance de la part externe de l'identifiant se fait comme pour l'alinéa précédent (cf exemple en I.2.I ci-dessous). *Cas particulier : l'objet est identifiable à partir d'un seul identifiant externe : il existe alors une relation (0,1) à (1,1) entre cet objet et un objet déjà identifié : cf § I.2.J.a ci-dessous*
  - L'objet n'est pas identifiable : il faut alors ajouter un identifiant (un "n° de code") dans le dictionnaire des données et recommencer à partir de l'étape 1.2.B.

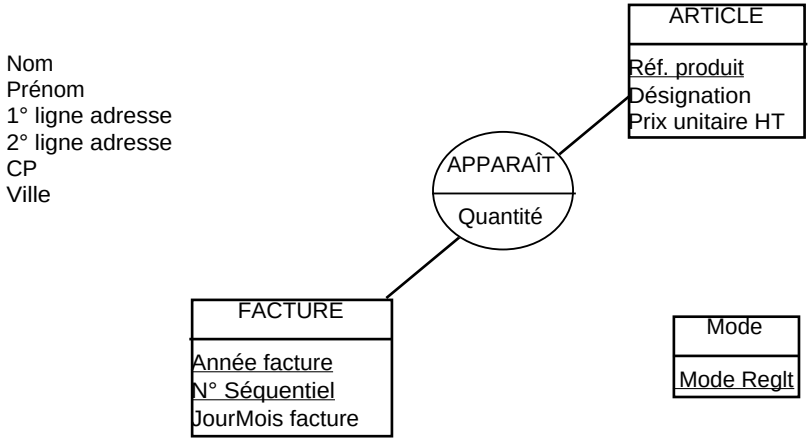
précautions :

- Ne créer d'identifiant supplémentaire dans le DDD que dans le cas où il est impossible d'identifier un objet ni en interne, ni en externe.
- Bien vérifier qu'il n'existe qu'une occurence d'un identifiant externe pour une occurence de l'objet.
- Dans le cas où plusieurs objets ne sont identifiables ni en interne, ni en externe, ne pas créer plus d'un identifiant à la fois dans le DDD (car un identifiant créé peut servir à identifier en externe d'autres objets que celui dans lequel il a été intégré).
- A chaque ajout d'identifiant, il est indispensable de refaire l'épuration du dictionnaire des données, car l'identifiant créé peut rendre caducs certains attributs initialement retenus.

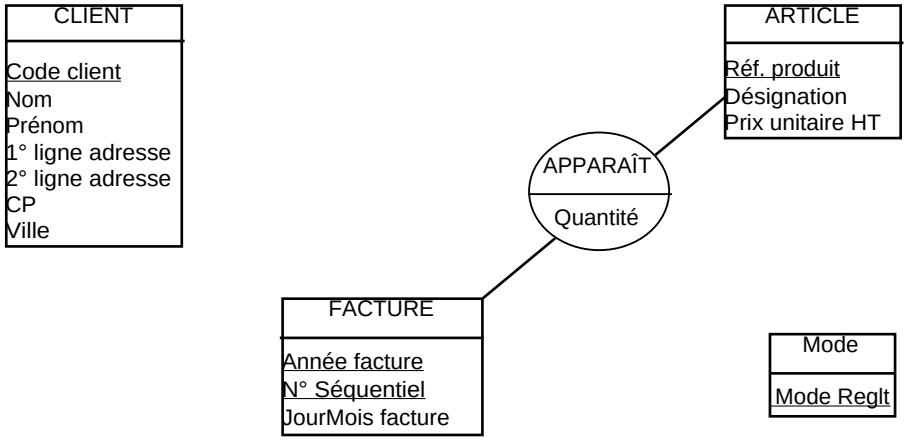
Remarques :

EXEMPLE :

Ebauche du M.C.D avant ajout du code client :




Ebauche du M.C.D après ajout du code client et reprise des étapes 1.2.B à 1.2.E (il n'y a pour l'instant pas de cas d'entité relative) :



## I.2.F - Définir les autres relations de dépendance entre les objets

But :
- Décrire l'existence d'autres relations, non porteuses de données, décrivant la dépendance, en particulier les contraintes d'existence, entre les entités.

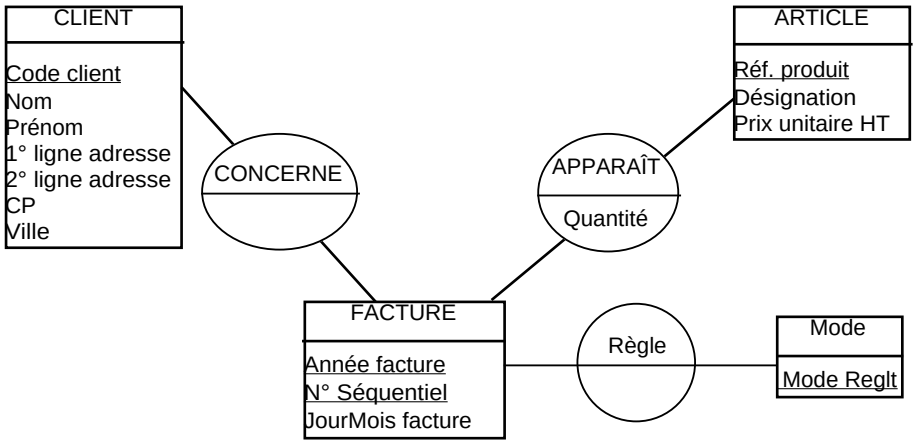
Moyens :
- Réaliser un tableau carré présentant en absisse et en ordonnée la liste des entités. Pour chaque case de ce tableau, déterminer les relations de dépendances susceptibles d'exister entre ce couple d'entités.
- Choisir un verbe pour représenter chaque relation reconnue. Placer la relation dans le M.C.D, et la relier à chacune des entités mises en jeu dans cette relation.

précautions :
- Lorsqu'on a mis en lumière l'existence d'une relation entre deux objets, vérifier s'il en existe une autre entre ces deux mêmes objets.
- S'il existe à la fois une relation entre un objet A et un objet B, entre B et C et entre A et C, vérifier :
  - si l'une des relations peut être une conséquence immédiate des deux autres. Dans ce cas, la supprimer.
  - si l'on est en présence d'une seule relation entre trois entités.
  - ou s'il existe bien deux manières différentes d'associer des occurences de C à des occurences de A, auquel cas il existera une boucle dans le MCD. *(NB : cette boucle nécessitera probabvement d'être exploitée par des requêtes faisant appel à des auto-jointures, par exemple :*

```
Select C1.Type, C2.Type, A.Nom
    From A, B, C C1, C C2
    Where B.CleA=A.Id
        And C1.CleB=B.Id
        And C2.CleA=A.Id)
```

Remarques :

EXEMPLE :



## I.2.G - Cardinalités

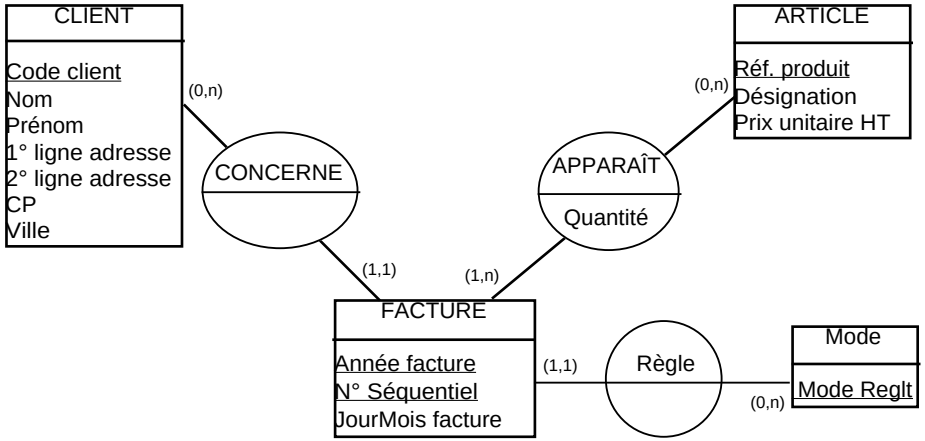
But :
- Décrire la nature de chaque relation.

Moyens :
- Etudier à tour de rôle chaque patte de chaque relation, c'est à dire chaque patte reliant une relation à une entité (ou une entité relative à une entité).
- Pour chaque patte, poser les deux questions :
  - Pour n'importe laquelle des occurences de l'entité, peut-il y avoir 0 occurences de la relation, ou doit-il y en avoir au moins une ?
  - Pour n'importe laquelle des occurences de l'entité, peut-il y avoir n occurences de la relation, ou doit-il y en avoir au plus une ?
- surmonter chaque patte du couple de réponses apportées : selon le cass, (0,1) ou (0,n) ou (1,1) ou (1,n).

précautions :
- Ne pas oublier les entités relatives.
- Ne pas pervertir les questions pour en faire par exemple "Pour n'importe laquelle des occurences de l'entité, peut-il y avoir 0 occurences *de l'autre entité (ou des autres entités) concourant à la relation*, ou doit-il y en avoir au moins une ?
- Ne pas oublier que souvent les cardinalités minimum ne se trouvent être que des indications de traitement, sans grande importance structurelle. Par contre les cardinalités maximum ont une imortance capitale dans l'estimation du poids du projet (ceci sera détaillé dans le passage au logique).
- En pratique : remettre en cause toutes les relations n'ayant aucun "n" comme cardinalité maximale (en théorie, ceci peut exister, mais en pratique, il y aura souvent intérêt à remplacer cette relation par la notion de sous-entité - voir exemples du cours). En conséquence, on pourra en pratique se permettre de négliger les cardinalités minimum. Une relation à deux pattes ayant une patte (0,1) ou (1,1) et une patte (0,n) ou (1,n) sera dite "relation 1-N". Une relation à deux pattes ayant deux pattes (0,n) ou (1,n) sera dite "relation N-N".
- Sauf cas très tatillons de relations ayant des cardinalités (0,1) à (0,1) remis en cause dans l'alinéa précédent, une relation porteuse de données aura toujours des cardinalités maximum de n sur toutes ses pattes (sinon, les attributs de la relation auraient pu être reportés dans l'entité reliée par cette patte). La réciproque n'est pas vraie : une relation non porteuse de données peut être du type 1-N ou N-N

Remarques :

EXEMPLE :



## I.2.H - Simplification à l'aide des contraintes d'intégrité fonctionnelle

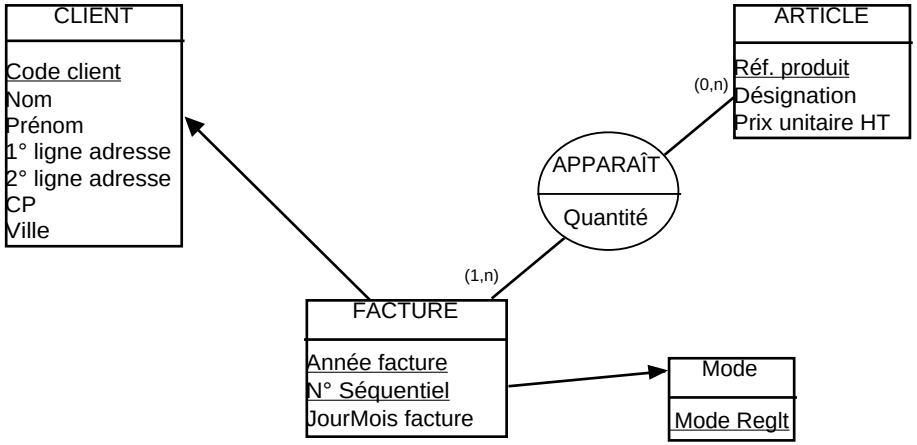
But :
- Modifier la présentation des relations les plus simples afin de représenter MIEUX et plus vite la complexité réelle du M.C.D.

Moyens :
- En théorie : Toute relation à deux pattes ayant sur une patte une cardinalité (1,1) sera remplacée par une simple flèche partant de l'entité reliée par la patte (1,1) et aboutissant à l'autre entité concourant à la relation. Chacune de ces relations est appelée "contrainte d'intégrité fonctionnelle", ou "CIF".
- En pratique : Toute relation à deux pattes ayant sur une patte une cardinalité (0,1) ou (1,1) sera remplacée par une simple flèche partant de l'entité reliée par la patte (0,1) ou (1,1) et aboutissant à l'autre entité concourant à la relation.

précautions :
- Ne pas oublier les entités relatives.
- Ne pas pervertir les questions pour en faire par exemple "Pour n'importe laquelle des occurences de l'entité, peut-il y avoir 0 occurences *de l'autre entité (ou des autres entités) concourant à la relation*, ou doit-il y en avoir au moins une ?.

Remarques :

EXEMPLE :



## I.2.I - une 9° étape : Vérification de la résistance au temps

But :
- S'assurer que le modèle obtenu a bien tenu compte des évolutions susceptibles d'évoluer dans le temps.

Moyens :
- Vérifier, pour chaque attribut de chaque objet, s'il y a conjonction des deux points suivants :
  - La valeur d'une occurence au moins de cet attribut peut être modifiée au cours de la vie du système.
  - Il faudra avoir accès à la fois à l'ancienne valeur et à la nouvelle valeur de cette occurence
- S'il existe des attributs pour lesquels la réponse à ces deux questions est simultanément positive, nous sommes en présence d'un M.C.D. qui "ne résiste pas au temps", et qui est donc basé sur un DDD erronné. Les corrections à apporter peuvent être de deux ordres :
  - Il existe une donnée cachée qui n'a pas été repérée dans le DDD (par exemple, la donnée "Date de modification" de l'attribut concerné"). Il faut alors ajouter cette donnée cachée dans le DDD et refaire tout le processus à pârtir de l'étape 1.2.B (en effet, l'ajout de ce nouvel attribut peut amener à épurer d'autres attributs devenus redondants).
  - Le nombre d'occurences de l'attribut a été sous-évalué : les valeurs différentes attribuées à cet attribut au cours du temps sont en fait des occurences différentes de cet attribut. Il faut alors reconsidérer le regroupement des attributs de l'étape 1.2.C

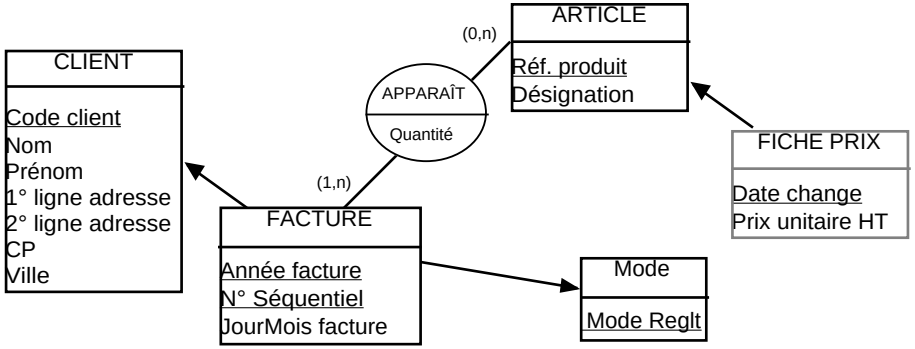
précautions :
- Il ne faut pas de remise en cause du M.C.D. obtenu à une réponse positive à seulement la première des deux questions (i.e. "La valeur d'une occurence au moins de l'attribut variera au cours du temps") : si une valeur change mais qu'on n'a pas besoin d'accéder à l'ancienne valeur, on a alors une seule occurence de l'attribut.

Remarques :

**1.**

EXEMPLE :

On veut gérer la résistance au temps du prix des articles (on considère arbitrairement que les changements de taux de TVA ne font pas partie de l'univers du discours) : ceci nous amène à introduire une donnée cachée (non visible dans les documents manipulés) "Date changement de prix d'article". L'étape 3 nous amène à créer un nouvel objet "Fiche Prix" qui contient les attributs "Prix unitaire HT" et "date de changement de prix". L'étape 5 nous amène à identifier cet objet à la fois avec le code article (externe) et la date (interne) : on a donc affaire à une entité relative.



## I.2.J - Cas particuliers

On peut, dans certaines études, se trouver confrontés à des cas “limites”, qui ne seraient pas traités d’une manière assez efficace par la méthode présentée ci-dessus :

### 1.2.J.a : Sous-entités :

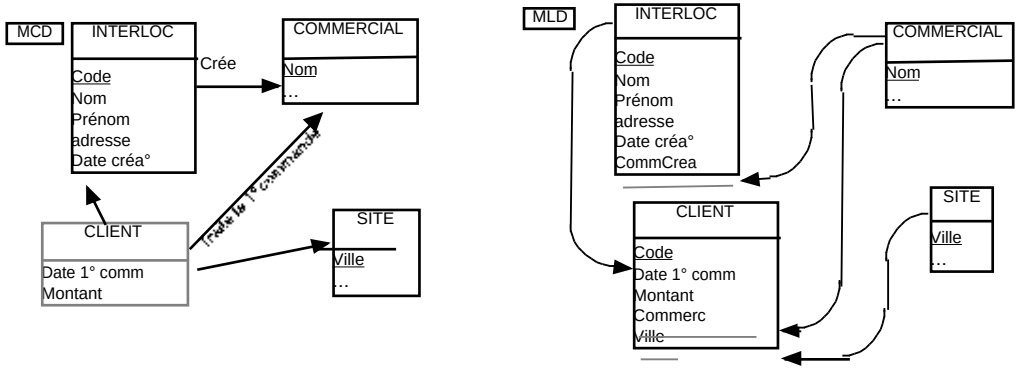
- Lorsqu’il existe entre deux entités une relation dont les cardinalités sont (0,1) à (1,1), on ne peut pas, en théorie regrouper ces deux objets en un seul (on ne peut faire ce regroupement que lorsque les cardinalités sont (1,1) à (1,1), et en principe on obtient un seul objet dès l’étape 3)
- Lorsqu’on a reconnu une entité E1, identifiée par un attribut A1, et qu’on a un objet non identifiable E2, dont l’identification est externe, et complètement réalisée en associant l’identifiant A1 à l’objet E2, on a alors une “entité relative sans identifiant interne”, et les cardinalités de la relation identifiante sont alors (1,1) à (0,1)

Dans ces deux cas, il est peu rentable de manipuler deux entités et une relation, alors qu’en fait il suffirait de regrouper malgré tout ces deux objets en un seul, en précisant que les attributs du 2° objet ne seront pas toujours renseignés (au prix d’une contrainte : Tous les attributs provenant de ce deuxième objets devront être simultanément renseignés ou non)
*Sur le terrain, je réalise systématiquement cette optimisation au niveau du MLD, mais ON NE PEUT PAS OPTIMISER UN MCD.*
C’est ici qu’on peut faire apparaître la notion de sous entité : on dit que E2 est une sous-entité de E1, sans identifiant. UN SGBD intégrant le concept d’héritage pourra manipuler les occurences de E1 qui ne correspondent à aucune occurence de E2 dans une table E1, et les occurences de E1 qui correspondent à une (et donc une seule) occurence de E2 dans une table enrichie E1+E2.
*Ceci correspond à peu près à la notion d’enregistrement avec partie variante du langage Pascal : Record ...Case...)*

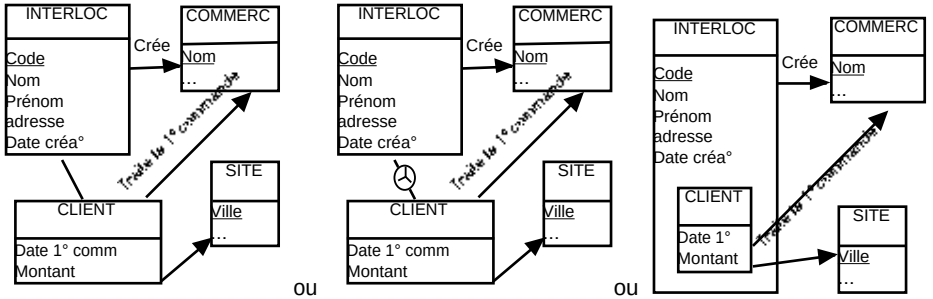
Remarques :

**1.**

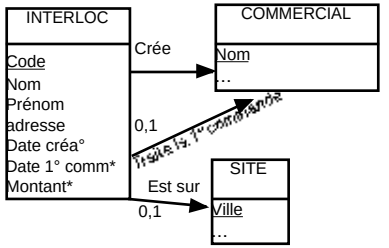
EXEMPLE :
On veut gérer les interlocuteurs d’une chaîne d’établissements commerciaux à des fins de mailing. Les commandes et factures ne sont pas gérées. Ces interlocuteurs sont des prospects ou des clients. Tous les clients commencent par etre des prospects. Pour chaque prospect, sont saisis les nom, adresse, tél, date de création, et est attribué un n° d’interlocuteur. Est également mémorisé le commercial qui a créé la fiche prospect. Lorsqu’un prospect devient client, on mémorise dans sa fiche ses coordonnées bancaires, la date de sa première commande, son montant, le site sur lequel cette commande a été passée, et le commercial l’ayant suivie. Une analyse rigoureuse de ces données fournirait les MCD et MLD suivants (cf pages 16 et suivantes pour la réalisation du MLD) :




Noter que le client est entièrement identifié par UN seul identifiant externe, d’où une entité relative sans élément d’identification interne. On constate dans le MLD que la jointure Interloc-Client est improductive. Une sous-entité se présentera de la manière suivante :




*Personnellement, c’est LA SEULE DERIVE QUE JE M’AUTORISE AU NIVEAU DU MCD :*




* : ces champs, si nuls, le seront simultanément. De plus, dans ce cas, les relations ”traite la 1° commande” et “Est sur” ne seront pas valorisées

**1.**

### 1.2.J.b : Contraintes entre relations :

Dans certains cas, on peut avoir des dépendances logiques entre deux relations distinctes : INCLUDE, AND et XOR.

par exemple, imaginons une relation R1entre deux entités E1 et E2, et une autre relation R2 entre deux entités E1 (à nouveau) et E3. Supposons qu'une occurence de R1 ne peut exister que si, pour la même occurence de E1, il n'existe aucune occurence de la relation R2 (cas du XOR) ou au contraire si l'occurence de R1 ne peut exister que s'il existe simultanément une occurence de R2 (cas du INCLUDE), ou si l'occurence de R1 ne peut exister que s'il existe simultanément une occurence de R2 et réciproquement (cas du AND)

On peut faire figurer ces contraintes dans un MCD

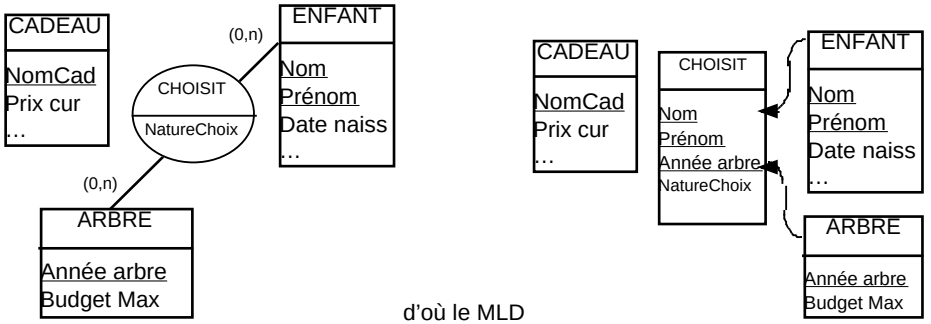
*NB : ceci ne concerne que les traitements, et je préfère de loin alléger mon MCD et surtout mon MLD en ne faisant pas figurer ce genre d'informations. En effet, ceci alourdit la lecture, sans aider à l'élaboration des requêtes. Pour moi, il faut donc limiter ces indications aux AGL proposant des applications générées qui mettront en œuvre les triggers ou les contraintes d'intégrité correspondant à ces restrictions logiques. Dans tous les cas, il faudra pouvaoir travailler au quotidien en exploitation sur un MLDdans lequel toutes ces informations sont filtrées.*

EXEMPLE :

## 1.2.J.c : Agrégats :

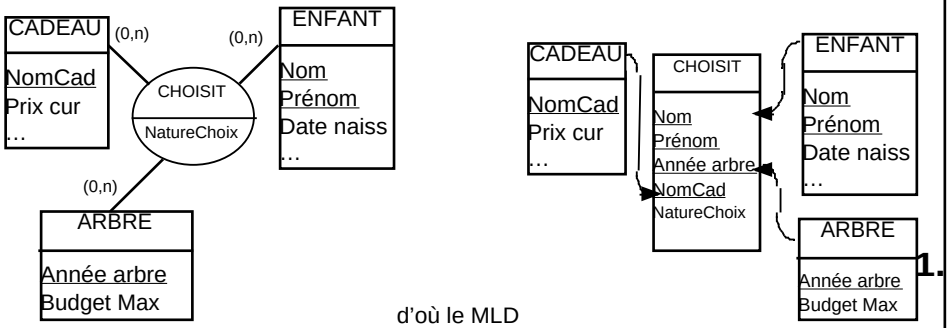
L'identification, au cours de l'étape 5, des relations porteuses de données, peut masquer des liens non identifiants entre cette relation et d'autres entités. La technique décrite ici ne permet pas de faire émerger ces liens. Mais la nature même de ces liens est différente des liens déjà présentés. Il faut donc pouvoir décrire des "pattes non identifianrtes" dans des relations multi-pattes.

EXEMPLE :
Pour un arbre de noël d'un comité d'entreprise, chaque enfant concerné reçoit chaque année un cadeau, choisi soit par le parent salarié, soit par le gestionnaire du système (donnée booléenne représentée par "Nature choix"). Les entités Enfant, Année et Cadeau ont déjà été identifiées (grâce à des éléments de l'univers du discours non reportés ici).
La technique présentée en étape 5 donne cette ébauche, la nature choix étant pleinement identifiée par l'enfant et l'année (un seul choix possible par enfant et par an, et une nature de choix et une seule par choix. D'autre part, on ne peut identifier le choix par Enfant et Cadeau, car le même enfant paut recevoir le même cadeau deux années différentes) :




d'où le MLD

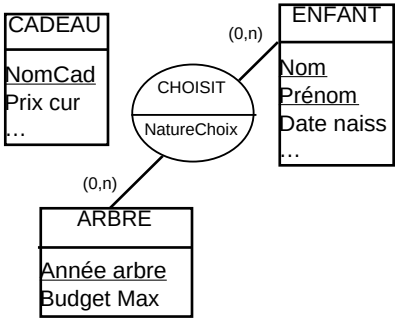
Cette représentation est incomplète, car il manque le fait qu'un choix concerne un cadeau. Mais la représentation suivante, qu'on peut pourtant facilement deviner, est fausse :



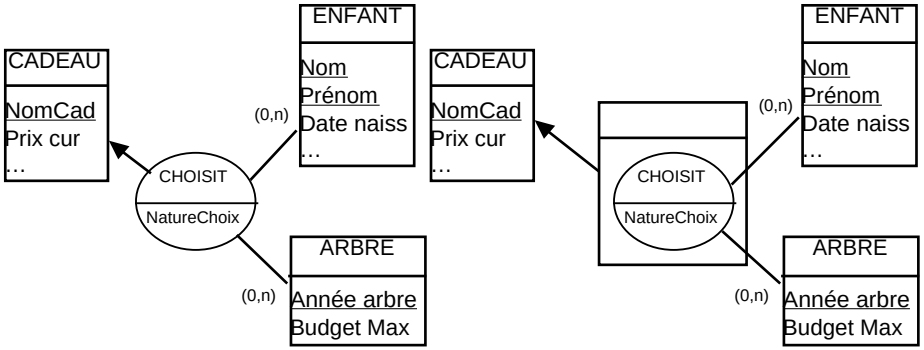

d'où le MLD

Dans ce modèle, on affirme qu'une natureChoix est identifiée par Enfant, Année ET cadeau. Ce système entraîne qu'un choix N'EST PAS PLEINEMENT IDENTIFIÉ par Enfant et Année, donc qu'un enfant ne peut pas avoir le même cadeau plusieurs fois la même année, mais il permet qu'il ait plusieurs cadeaux la même année, à condition qu'ils soient différents.

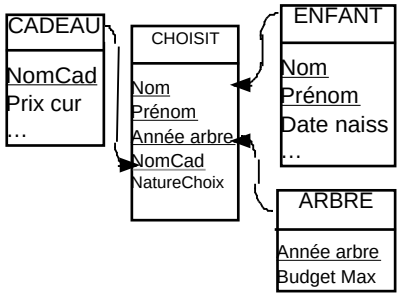
La solution consiste à conserver les seules pattes identifiantes à l'étape 5 :




Puis, à l'étape 6, à vérifier s'il existe des liens entre les objets du MCD, que ces objets soient des entités, des entités relatives, OU des relations porteuses de données. S'il existe un lien entre une relation porteuse de données et une entité (ou une entité relative), il faudra utiliser l'une des deux représentations suivantes.




qui fourniront le MDL suivant :




Ce MLD ressemble beaucoup au 2° MLD de la page précédente. La différence est pourtant importante : NomCad ne fait plus partie de la clé primaire de CHOISIT, et le risque de doublons décrit plus haut n'existe plus.
*NB1 : ce genre d'erreurs ne se révèle souvent que longtemps après la mise en service, il est donc coûteux en maintenance, car il est parfois difficile de se plonger dans d'anciens développements.*
*NB2 : Certains designers (Win'Design par exemple) ne savent pas représenter ces liens. Il faudra donc retoucher manuellement le MLD pour obtenir un système pérenne.*

# II - La réalisation d'un M.L.D. standard

## II.1 - Ce qu'on attend d'un M.L.D. standard

But :

- Il s'agit de représenter, par un formalisme précis et standardisé, l'ensemble des tables qu'il faudrait créer pour réaliser le projet décrit dans le M.C.D, dans le cas où l'on aurait à disposition une machine, des équipes de développement et des outils de programmation de puissance et de capacité infinies.

- L'adaptation à l'environnement concret sera fait en aval de cette opération. Ainsi, tout changement de système, d'équipe ou d'outils de développement pourra s'appuyer sur le M.L.D. sans remettre en cause le travail réalisé en amont.

# II.2 - Le passage au logique.

## II.2.A - La transformation des entités

But : • Il s'agit de déterminer les tables nécessaires au stockage des informations relatives à une entité.

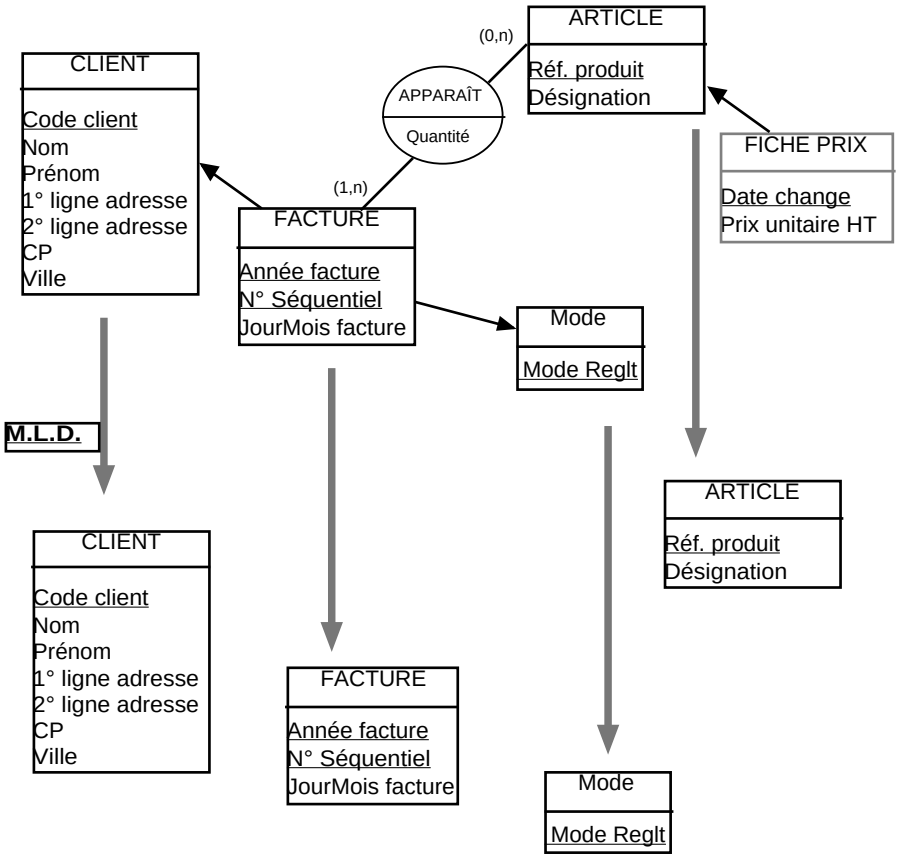
Moyens : • En langage rigoureux : Une entité est représentée par une table, dont le nom est le même que le nom de l'entité, dont les colonnessont en correspondance bi-univoque avec les attributs de l'entité et en récupèremnt les noms, et dont la clé primaire est composée de la concaténation des colonnes correspondant aux attributs participant à l'identifiant.
• En pratique : Une entité devient une table. Un identifiant devient une clé primaire.

précautions : • Il faut ici admettre que le système "parfait" susceptible d'accueillir ce M.L.D. permet de réaliser un index sur plusieurs colonnes.

Remarques :

EXEMPLE :

**M.C.D.**




**M.L.D.**

## II.2.B - La transformation des relations 1-N

But :
- Il s'agit de déterminer les colonnes et liaisons nécessaires au stockage des informations relatives à une relation 1-N.

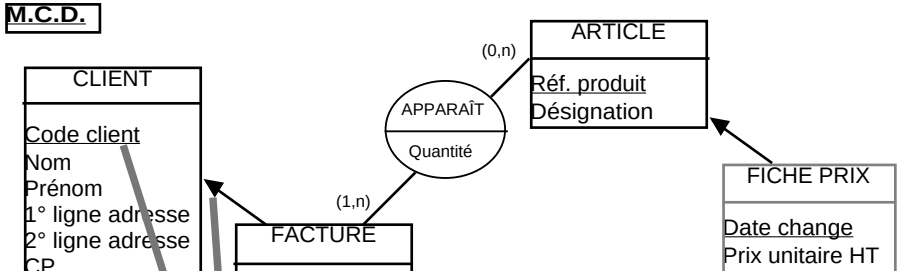
Moyens :
- Une relation 1-N est représentée par deux éléments :
  - La création d'une colonne dans la table découlant de l'entité située du côté 1 de la relation.Cette colonne est composée de l'identifiant de l'entité située du côté N de la relation. Cette colonne est dite "clé étrangère" , elle sera soulignée en pointillés.
  - une liaison entre l'intitulé de la table découlant de l'entité située du côté N de la relation et cette clé étrangère.

précautions :
- Si l'identifiant de l'entité située du côté N de la relation est composé de la concaténation de plusieurs attributs, la partie de l'identifiant récupérée dans la table qui découle de la relation le sera aussi.
- Attention : lesens des flèches du M.L.D est inversé par rapport au sens des flèches des CIF du MCD.

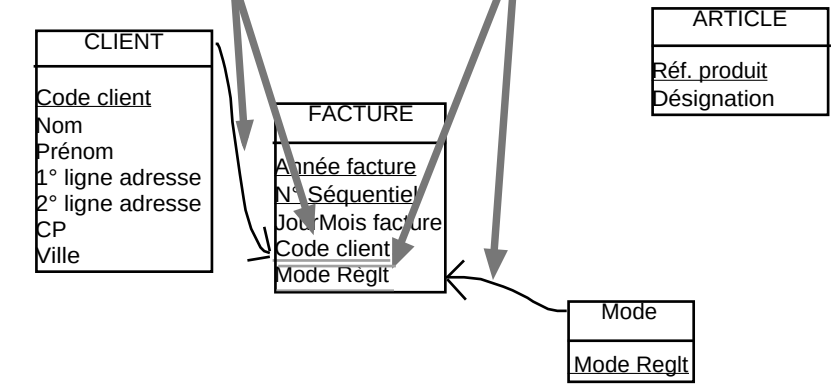
Remarques :

EXEMPLE :

**M.C.D.**




**M.L.D.**



## II.2.C - La transformation des relations N-N

But :
- Il s'agit de déterminer les tables nécessaires au stockage des informations relatives à une relation N-N.

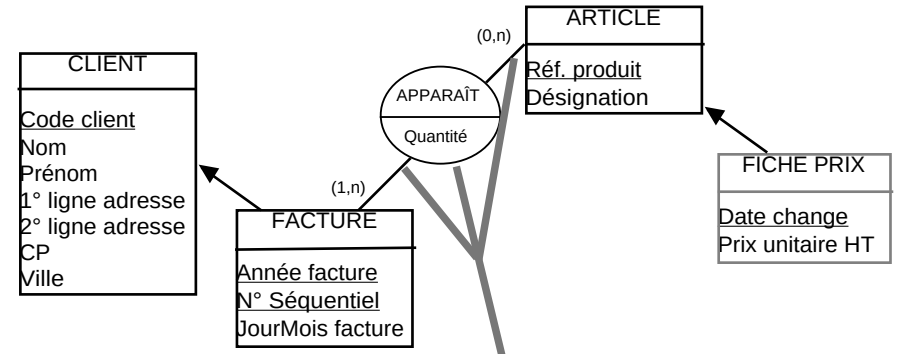
Moyens :
- Une relation N-N est représentée par trois éléments :
  - La création d'une table contenant les attributs portés par la relation (s'il y en a), à laquelle on ajoute les identifiants des entités concourant à la relation. La concaténation de ces identifiants fournit la clé primaire de cette table. Le nom de la relation (qui est un verbe) est souvent remplazcé par un nom mieux adapté à une table.
  - une liaison entre l'intitulé des tables découlant des entités concourant à la relation et la partie de l'identifiant de la table découlant de la relation.

précautions :
- Si l'identifiant d'une entité N est composé de la concaténation de plusieurs attributs, la partie de l'identifiant récupérée dans la table qui découle de la relation le sera aussi.
- Si la relation a plus de deux pattes, on procède de même avec chacune des pattes ayant une cardinalité maximale égale à n.
- L'ordre dans lequel les identifiants des entités sont récupérés pour fournir la clé unique de la table découlant de la relation n'a pas d'importance structurelle. Il s'agira tout au plus d'une optimisation dans la taille ou l'efficacité des index en découlant (voir exemples du cours).

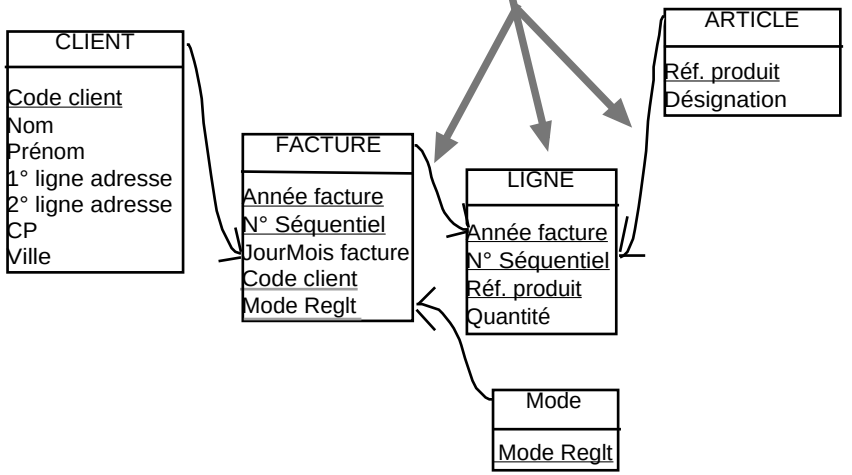
Remarques :

EXEMPLE :

**M.C.D.**




**M.L.D.**



## II.2.D - La transformation des entités relatives

But :
- Il s'agit de déterminer les attributs et tables nécessaires au stockage des informations relatives à une entité relative.

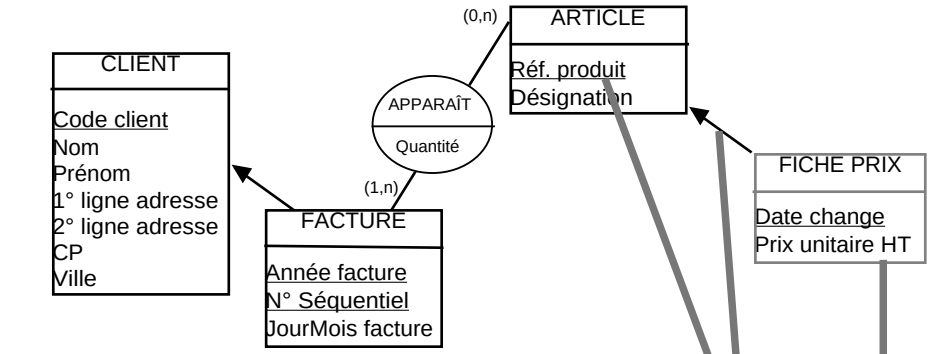
Moyens :
- Une entité relative est représentée comme une entité et une relation. Pour cela, il faut réaliser tois actions :
  - La création d'une table découlant de l'entité relative.
  - La création d'une clé étrangère, comme pour toute relation, c'est-à-dire l'ajout de l'identifiant de l'entité pointée par l'entité relative dans la table découlant de cette entité et de la liaison entre la table découlant de l'entité pointée et cette clé étrangère.
  - L'utilisation de cette clé étrangère comme composant de l'identifiant de la table découlant de l'entité relative.

précautions :
- Si l'identifiant d'une entité N est composé de la concaténation de plusieurs attributs, la partie de l'identifiant récupérée dans la table qui découle de la relation le sera aussi.
- Si la relation a plus de deux pattes, on procède de même avec chacune des pattes ayant une cardinalité maximale égale à n.
- L'ordre dans lequel les identifiants des entités sont récupérés pour fournir la clé unique de la table découlant de la relation n'a pas d'importance structurelle. Il s'agira tout au plus d'une optimisation dans la taille ou l'efficacité des index en découlant (voir exemples du cours).

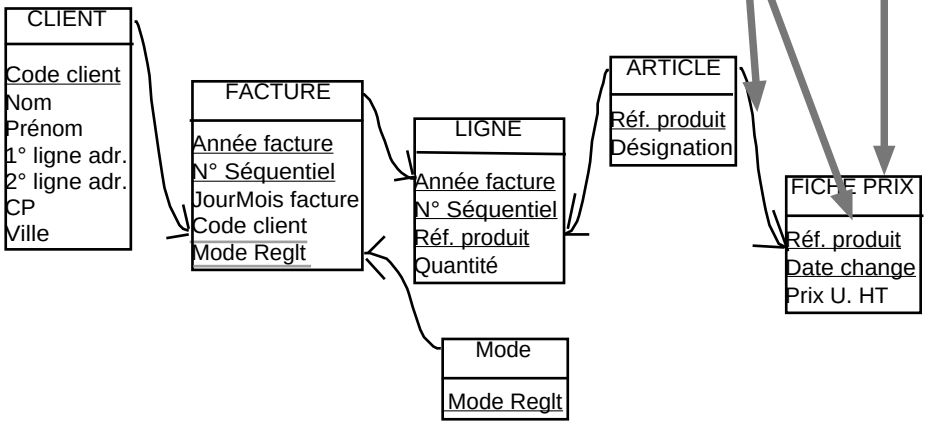
Remarques :

EXEMPLE :

**M.C.D.**




**M.L.D.**



# III - La réalisation d'un M.L.D. optimisé

## III.1 - Ce qu'on attend d'un M.L.D. optimisé

But :

- Il s'agit de définir les modifications à apporter au M.C.D. afin de le rendre mieux adapté à l'environnement concret dans lequel il sera implémenté.

## III.2 - Optimisations systématiques

- Toutes les colonnes dont le nombre de lignes est au plus égal à 1 seront regroupées dans une table de paramètres.
- Les tables dont le nombre de colonnes est égal à 1, et qui sont cibles de liens (dans la représentation CODASYL du M.L.D.) sans être origines de liens seront épurées (ces tables correspondent en général à des attributs pouvant prendre un nombre fini et prédéterminé de valeurs distinctes. Ces tables peuvent souvent être épurées dès le M.C.D. voir exemple, table "Mode" ).
- On placera systématiquement un index sur la clé primaire de chaque table.

## III.3 - Optimisations courantes

- Dans un S.G.B.D. acceptant les valeurs 'NULL', les tables découlant des sous-entités seront intégrées dans les tables découlant de l'entité principale.
- On envisagera de placer des index sur les clés étrangères. En particulier, lorsque des écrans interactifs font apparaître des 'listes incluses', il sera important d'indexer les colonnes représentant la clé étrangère.
- Dans un S.G.B.D. ne permettant pas la définition d'un index sur plusieurs colonnes, il y aura création d'une colonne calculée comme concaténation des colonnes à indexer. Si possible, cette règle de calcul sera définie par un trigger. C'est cette colonne calculée qui sera indexée.
- Remettre en cause les clés composées dont les composants discrets ne représentent pas d'intérêt (c.f. exemple : le découpage année Facturation+n° séquentiel est finalement peu intéressant).



## III.4 - Pistes pour d'autres optimisations

*TOUTES LES OPTIMISATIONS PROPOSÉES DANS CE PARAGRAPHE PEUVENT ÊTRE REMISES EN CAUSE.*

- Les tables issues de relations N<->N ont toujours une clé primaire composite, dont chaque élément est aussi une clé étrangère. Il faudra donc indexer la concaténation des colonnes identifiantes, mais il sera aussi souvent nécessaire de'avoir un accès indexé aux clés étrangères . Il sera intéressant de ne pas indexer la première des clés étrangères concaténées dans la clé primaire, et de se servir de la clé primaire comme de cette clé étrangère en négligeant la fin de cette clé (voir exemple).
- Eviter les colonnes calculées à partir d'autres colonnes de la même table, sauf en cas d'index : en général, les temps de calcul seront toujours négligeables devant les temps d'accès.
- Il est en général peu intéressant de créer une colonne redondante pour éviter l'éxécution une boucle : par exemple, il est peu intéressant de mémoriser un cumul d'une colonne dans une table liée directement, car le recalcul de ce cumul est généralement assez rapide (par exemple, pour visualiser une facture, le montant total d'une facture n'a pas besoin d'être mémorisé dans la table Facture).
- Il est en général intéressant de créer une colonne redondante pour éviter l'éxécution de deux boucles imbriquées : par exemple, il est intéressant de mémoriser un cumul d'une colonne dans une table liée directement, lorsque ce total doit être affiché dans chaque ligne d'une liste utilisée fréquemment (par exemple, pour visualiser la liste des factures, le montant total d'une facture doit être mémorisé dans la table Facture si on veut un affichage fréquent de la liste des factures). En particulier, si une requête doit être lancée à lintérieur d'une boucle, les temps de réponse seront contraignants.
- Lorsqu'une entité estliée à deux autres par deux liens N à N, envisager la création d'une table redondante de liens croisés.
- Etudier l'intérêt d'index sur les identifiants utilisateur (nom, désignation) ou sur les codes externes (code postal).
- Etudier les optimisations réseau et les tables réparties :
  - Les optimisations liées à l'organisation, découpage vertical (certaines tables sur un site, d'autres sur un autre).
  - Les optimisations liées à la charge, découpage horizontal (certaines lignes d'une table sur un site, d'autres sur un autre).
- Envisager la suppression de tables d'énumération et de les remplacer par des fonctions (codage en dur), ou chargement de ces tables dans des tableaux de variables (initialisations)