



CONCEPTION DE BASES DE DONNÉES

(RELATIONNEL/MERISE/UML)

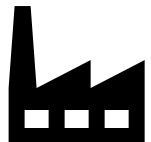


MERISE

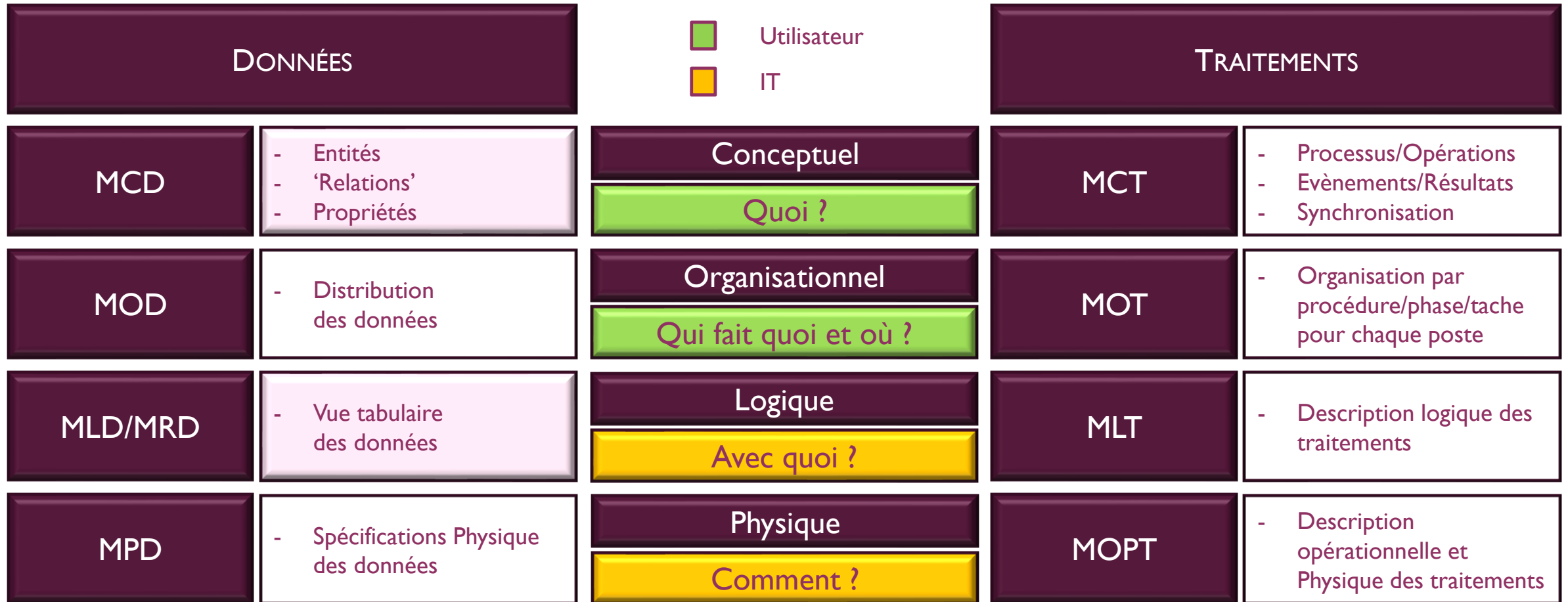
Vu de la Donnée...

MÉTHODE MERISE – ORIGINE

- Méthode d'analyse, de conception et de gestion de projets informatiques
 - Emerge vers la fin des années 1970 sous l'impulsion de René Colleti, Arnold Rochfeld et Hubert Tardieu, alors consultants et manager de SSI françaises (SEMA, CGI Informatique).
 - Surtout utilisée en France sur les grands projets en tant que méthode générale.
- Caractérisé par :
 - Une approche systémique de l'entreprise,
 - Une approche parallèle et conjointe concernant les données et les traitements
 - Une approche par niveau (conceptuel, organisationnel, logique, physique)



MÉTHODE MERISE – APPROCHE PAR NIVEAUX



MÉTHODE MERISE – MCD

- Permet de décrire de manière formelle les données manipulées par le SI
- L'univers du MCD est constitué des éléments suivants :
 - Les Entités
 - Les Propriétés,
 - Les 'Relations' ou Associations.

MÉTHODE MERISE – MCD – ENTITÉS – INTRODUCTION (I)

■ Entité : définition(s)

Difficile de trouver une définition formelle on trouve de tout dans la littérature !

- « Regroupements d'informations qui possèdent des attributs (caractéristiques) »
- « (Les entités) sont définies par certaines propriétés (numéro, nom, prénom...) que l'on est amené à regrouper naturellement ».
- « Objet pourvu d'une existence propre et conforme aux choix de gestion de l'entreprise »

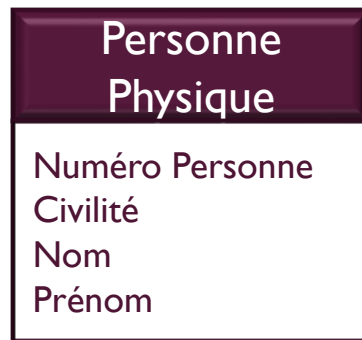
- La chose 'Client' suivante est-elle une entité :



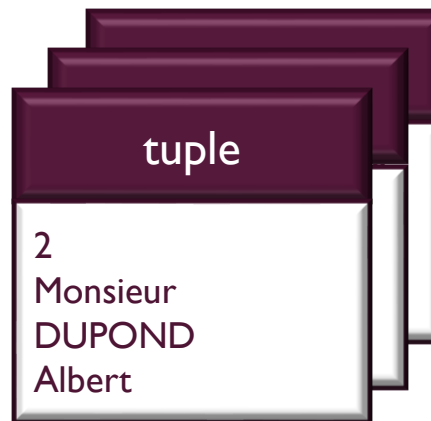
MÉTHODE MERISE – MCD – ENTITÉS – INTRODUCTION (3)

■ Entité : occurrences et tuples

L'entité est le concept, l'occurrence est un élément de l'entité, l'ensemble des valeurs d'attributs/propriétés d'un occurrence forme un tuple



Entité



Occurrences
de l'Entité

MÉTHODE MERISE – MCD – ENTITÉS - PROPRIÉTÉS (I)

■ Propriétés : définition(s)

On peut lire :

« une propriété est une donnée élémentaire perçue sur une entité ou une relation. »

« Regroupements d'informations qui possèdent des attributs (caractéristiques) »

Le plus souvent le concept est introduit par un exemple :

« ..

Par exemple, si l'on considère l'entité "être humain" les informations communes aux être humains peuvent être :

- le nom
- le ou les prénoms
- la date de naissance
- le lieu de naissance
- le sexe
- l'adresse

»

MÉTHODE MERISE – MCD – ENTITÉS - PROPRIÉTÉS (2)

■ Type d'une propriétés

On peut lire ici aussi :

« On devra préciser le type des données attendues pour chaque attribut »

Par 'attribut' l'auteur(bien connu) de cette injonction entend en fait 'propriété' et par type de donnée quelque chose comme « char(16), smallint, float,... »

- En fait la notion de type de la propriété conduit bien vite à de la sur-spécification ! Nous nous prétendons au niveau conceptuel et il faut préciser la taille en octets d'un nom ?
- Merise passe ici complètement à côté de la notion de « domaine » qui sera révélée par T. CODD et le relationnel (le vrai....)

Dans le suite de l'exposé certains concepts, venant du relationnel, inexistant formellement dans MERISE, mais nécessaires pour la production de MCD plus carrés, seront transposés dans MERISE

MÉTHODE MERISE – MCD – ENTITÉS – PROPRIÉTÉS (3)

■ Domaine - Concept non MERISE (I)

Un Domaine est un ensemble de valeurs **sémantiquement cohérentes** que l'on peut vouloir ou non comparer entre elles.

Ex : domaine des 'factures clients'

- Un domaine peut être constitué d'une liste de valeurs, d'intervalles, de types de base (entier, date, réel...),
- Une propriété MERISE, mais MERISE l'ignore, fait en fait référence à un domaine,
- Le modèle relationnel s'appuie fondamentalement sur la notion de domaine qui permet de développer des modèles de données sémantiquement « carrés »

MÉTHODE MERISE – MCD – ENTITÉS – PROPRIÉTÉS (4)

■ Domaine - Concept non MERISE (2)

Domaine des
Dates de Factures



Domaine des
Dates de Factures
Clients



Domaine des
Dates de Factures
Fournisseurs



- On ne doit pouvoir comparer que ce qui est comparable.
- Mélanger le domaine des ‘dates de factures clients’ avec celui des ‘dates de factures fournisseurs’ n’a guère de sens.
- Oublier cela, **faire en sorte que les domaines manipulés soient des collections hétéroclites, conduira implacablement à un modèle peu exploitable et peu performant** (François de Sainte Marie).
- Les domaines sont des connecteurs sémantiques pour les (*futures*) relations (Serge MIRANDA 1988).

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (I)

■ Principe

On lit : « Une des propriétés a un rôle bien précis, c'est l'identifiant nommé aussi la clé. L'identifiant permet de connaître de façon sûre et unique l'ensemble des propriétés qui participent à l'entité. »

En fait :

- La clé peut être constituée de plusieurs propriétés
- Il peut exister plusieurs clés candidates éligibles en tant qu'identifiant...

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (2)

■ Clés candidates (coup de pouce du relationnel)

Une clé candidate est un ensemble $C_c = \{A_x, A_y, \dots, A_z\}$ de propriétés/attributs d'une 'Entité' (dans le relationnel on dirait entête de relvar) vérifiant les principes d'**unicité** et de **minimalité**.

- **Unicité**

Soit t_1 et t_2 deux tuples distincts quelconques de R . Si C_{c1} et C_{c2} sont les valeurs prises par C_c respectivement pour t_1 et t_2 , alors on doit invariablement vérifier l'inégalité : $C_{c1} \neq C_{c2}$

- **Minimalité**

L'unicité n'est plus maintenue si on supprime de C_c un des attributs A_x, A_y, \dots, A_z .

Toute entité doit disposer d'au moins une clé candidate.

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (3)

■ Dépendance Fonctionnelle

La notion de **D**épendance **F**onctionnelle (notée après DF) permet de déterminer toutes les clés candidates d'une entité

Soit X et Y deux sous-ensembles d'attributs, non nécessairement disjoints, d'une entité. On dit que **Y dépend fonctionnellement de X** et que **X détermine Y** si, pour toute valeur de X , il ne peut y avoir qu'une valeur unique de Y .

En d'autres termes, si 2 tuples ont la même valeur pour X alors ils ont nécessairement la même valeur pour Y .

On note : $X \rightarrow Y$ X est le **déterminant** de la DF et Y le **dépendant**.

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (4)

■ Types de Dépendance Fonctionnelle

- Dépendance Fonctionnelle triviale

Soit X et Y deux sous-ensembles d'attributs d'une entité. La DF $X \rightarrow Y$ est triviale si Y est inclus dans X

- Dépendance Fonctionnelle partielle

Soit X un sous-ensembles d'attributs d'une entité E et A_i un attribut quelconque de E . La DF $X \rightarrow \{A_i\}$ est partielle si :

- Elle n'est pas triviale,
- Il existe Y strictement inclus dans X tel que $Y \rightarrow \{A_i\}$

- Dépendance Fonctionnelle irréductible à gauche (ou élémentaire ou totale)

Une DF est dite irréductible à gauche si elle n'est ni triviale, ni partielle

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (5)

■ Sur-clé

- Définition

Une sur-clé est un sous-ensemble SC d'attributs d'une entité E respectant la contrainte d'unicité mais pas celle de minimalité

⇒ Quelque soit l'attribut $\{A_i\}$ de E, $SC \rightarrow \{A_i\}$

■ Sous-clé

- Définition

Une sous-clé est un sous-semble sC, non nécessairement strict, d'une clé candidate Cc d'une entité E.

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (6)

■ Les 3 axiomes d'Armstrong

Ces axiomes permettent de déduire l'ensemble des DF d'une Entité à partir des DF connues.

Soit A, B, C, D des sous-ensembles d'attributs d'une Entité.

- **Réflexivité** : si $A \supseteq B$ alors $A \rightarrow B$
- **Augmentation** : si $A \rightarrow B$ alors $AC \rightarrow BC$
- **Transitivité** : si $A \rightarrow B$ et $B \rightarrow C$ alors $A \rightarrow C$

De ces 3 axiomes on déduit :

- **Union** : si $A \rightarrow B$ et $A \rightarrow C$ alors $A \rightarrow BC$
- **Pseudo-transitivité** : si $A \rightarrow B$ et $BC \rightarrow D$ alors $AC \rightarrow D$
- **Décomposition** : si $A \rightarrow B$ et $B \supseteq C$ alors $A \rightarrow C$
- **Composition** : si $A \rightarrow B$ et $C \rightarrow D$ alors $AC \rightarrow BD$

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (7)

■ Fermeture des DF

Si F est un ensemble de Dépendances Fonctionnelles d'une entité, on appelle fermeture de F , que l'on note F^+ , l'ensemble des DF que l'on peut déduire de F par application des axiomes d'Armstrong et de leurs règles déduites.

■ Couverture minimale

Ensemble minimum de DF appartenant à F à partir duquel on peut obtenir F^+

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (8)

■ Dépendances Fonctionnelle – Illustration (I)

- Règles (un peu farfelues...) :
 - (R1) Les Salariés appartiennent à un groupe composé de plusieurs sociétés
 - (R2) 2 Salariés ne peuvent avoir le même matricule dans tout le groupe
 - (R3) Un Salarié d'un matricule donné ne fait partie que d'une seule société, n'a qu'un nom et qu'une seule date d'embauche.
 - (R4) Un Salarié qui a quitté le groupe peut être réembauché. Il dispose alors d'un nouveau matricule
 - (R5) Deux Salariés portant le même nom ne peuvent être embauchés la même année

SALARIÉ

matricule
nom
code_societe
date_embauche

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (9)

■ Dépendances Fonctionnelle – Illustration (2)

- DF déduites des règles :

- (DF1) : {matricule} → {nom} (R3)
- (DF2) : {matricule} → {code_societe} (R3)
- (DF3) : {matricule} → {date_embauche} (R3)
- (DF4) : {nom, date_embauche} → {matricule} (R5)

$$F = \{DF1, DF2, DF3, DF4\}$$

On déduit par transitivité la DF à intégrer dans F^+ :

- (DF5) : {nom, date_embauche} → {code_societe}

SALARIÉ

matricule
nom
code_societe
date_embauche

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (10)

■ Identifiant MERISE - Principe

Voici une définition que l'on peut trouver :

« L'identifiant d'une entité est une propriété particulière telle qu'à chaque valeur de cette propriété corresponde une et une seule occurrence de l'entité. Une entité possède un et un seul identifiant »

- Dans Merise, il semblerait que ce rôle de clé soit assigné à une seule propriété
- En réalité, si l'on s'imprègne de culture relationnelle, ce rôle d'identifiant peut être confié à l'une des clés candidates
 - Il convient de choisir l'une des clés candidates comme identifiant.
 - Remarque : si SQL impose l'existence d'un tel identifiant, appelé clé primaire, le modèle relationnel n'en a pas vraiment besoin...

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (II)

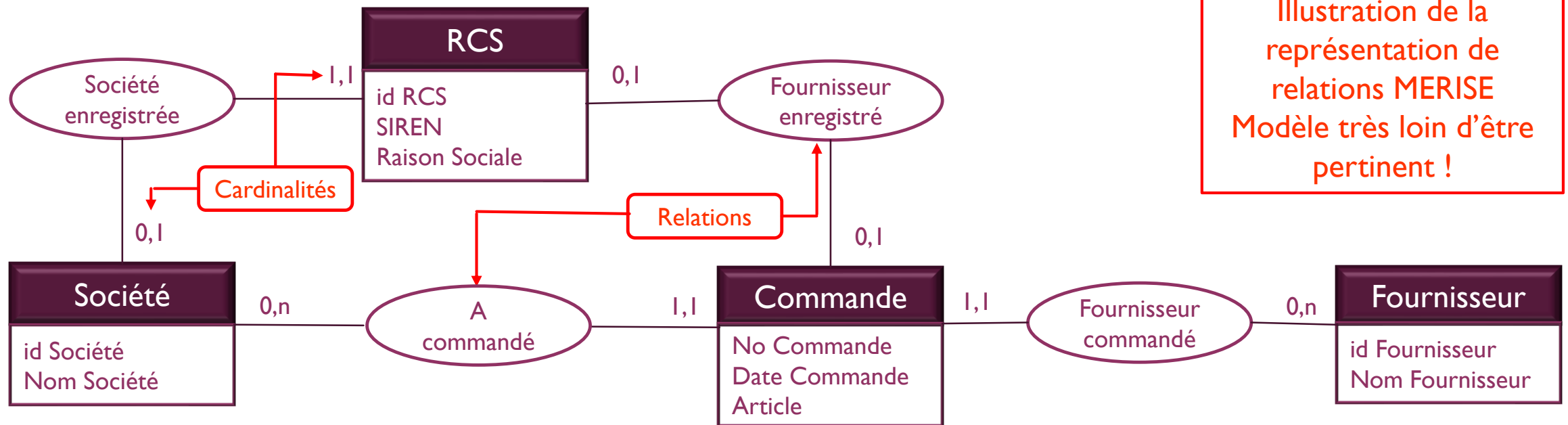
■ Identifiant MERISE – Choix de la clé

- La clé candidate id doit satisfaire à certaines règles pour devenir identifiant (ou, disons le clé primaire) :
 - Les valeurs de id ne peuvent contenir des données confidentielles
 - Les données de id doivent être stables et ne doivent pas dépendre d'un organisme ou système extérieur à l'entreprise.
 - ⇒ On ne peut pas choisir un numéro SIREN comme identifiant d'un client car :
 - ❖ Les sociétés peuvent changer de SIREN,
 - ❖ La structure d'un numéro SIREN peut être amenée à changer.
 - L'identifiant id ne doit pas porter des données significatives car la sémantique de données significative est susceptible d'évoluer dans le temps.
- Si aucune des clés candidates ne satisfait à ces règles, on en invente une, 'artificielle', en général un entier (non signé !) que l'on incrémente au fur et à mesure des insertions.

MÉTHODE MERISE – MCD – RELATIONS/ASSOCIATIONS (I)

■ Relation MERISE - Principe

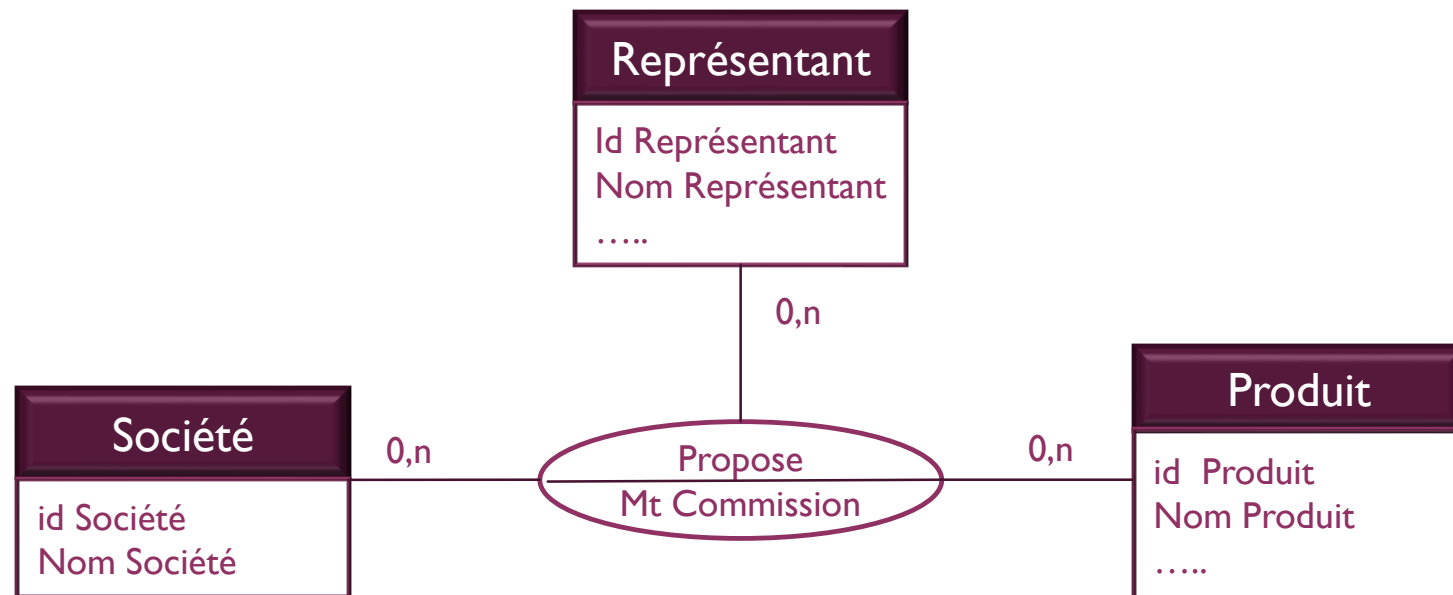
Une relation (ou association) MERISE est une association perçue dans le réel entre deux ou plusieurs entités. Une relation n'a pas d'existence propre.



MÉTHODE MERISE – MCD – RELATIONS/ASSOCIATIONS (2)

■ Relation MERISE - Dimension

La dimension d'une relation est le nombre d'entités participant à cette relation .



Exemple de relation
ternaire porteuse

MÉTHODE MERISE – MCD – TYPES D'ENTITÉS

■ Entités fortes

Une entité de ce type a une existence propre. Elle s'oppose à ce que toute disparition d'une entité avec laquelle elle entretient des relations provoque sa propre disparition.

Un client, un fournisseur, un pays,...

■ Entités faibles

C'est une propriété multivaluée d'une entité plus forte qu'elle. Ces entités n'ont pas d'existence propre et disparaissent lorsque l'entité 'parente' disparaît.

Typiquement l'entité 'ligne de facture', propriété multivaluée de l'entité 'facture'

MÉTHODE MERISE – MCD – ENTITÉS – IDENTIFIANT (12)

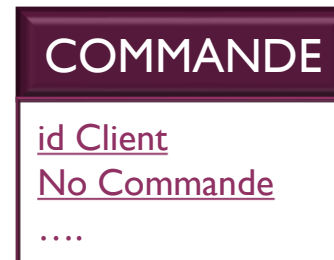
■ Identifiant MERISE – Identifications absolues et relatives

Cette question concerne les Entités faibles. Prenons le cas des entités CLIENT (forte) et COMMANDE (faible, en effet, une commande n'existe pas sans client)



Absolute : Numéro absolu incrémenté à chaque nouvelle commande

OU ?

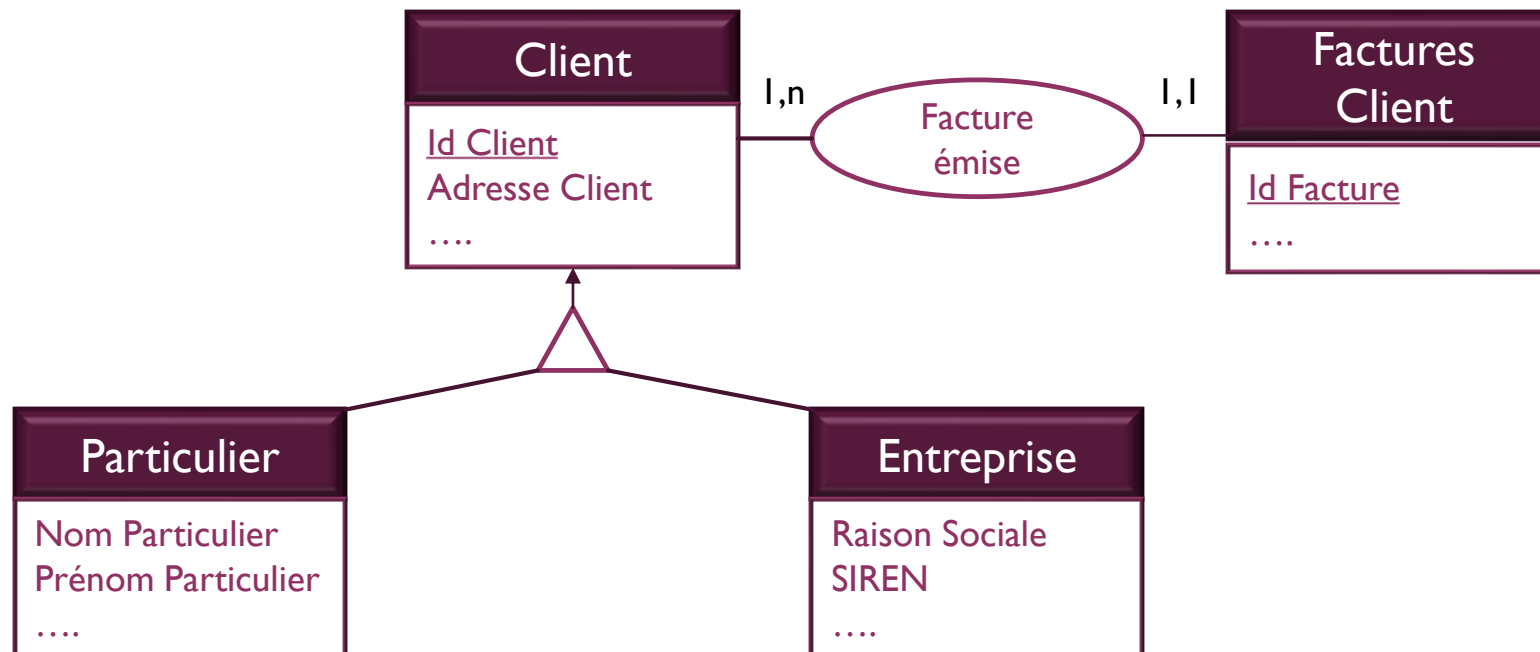


Relative : Pour chaque Client, numéro incrémenté à chaque nouvelle commande. Tous les Clients qui ont commandé disposent d'une commande avec No Commande = I

MÉTHODE MERISE / 2 – MCD – ENTITÉS - HÉRITAGE

■ MERISE / 2 Généralisation / Spécialisation (Héritage)

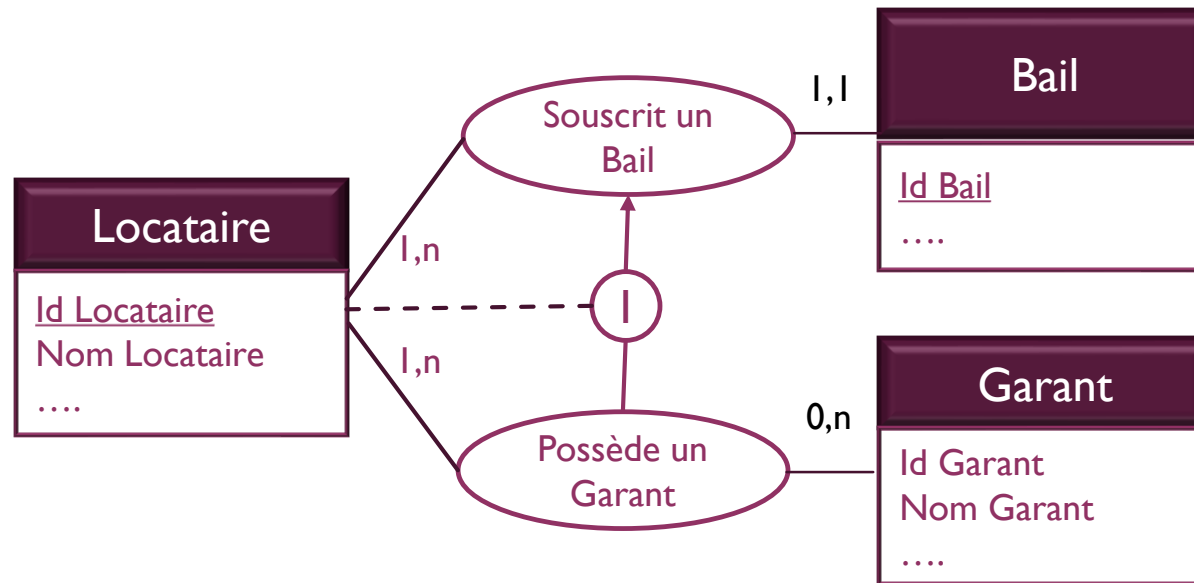
La Spécialisation permet de définir des entités enfants, intégrant des propriétés spécialisées, héritant des propriétés d'une entité par



MÉTHODE MERISE / 2 – MCD – CONTRAINTES (I)

■ MERISE / 2 Contrainte d'inclusion

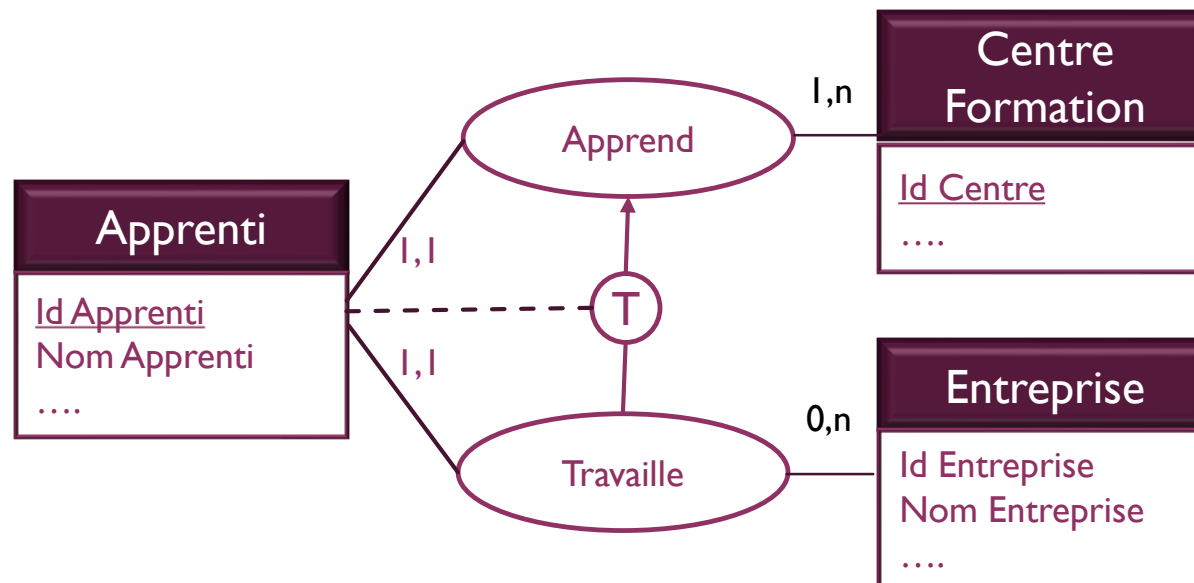
Une contrainte d'inclusion implique que pour une entité participant à deux relations, toute occurrence de l'entité participant à la première relation doit participer également à la deuxième relation.



MÉTHODE MERISE / 2 – MCD – CONTRAINTES (2)

■ MERISE / 2 Contrainte de totalité

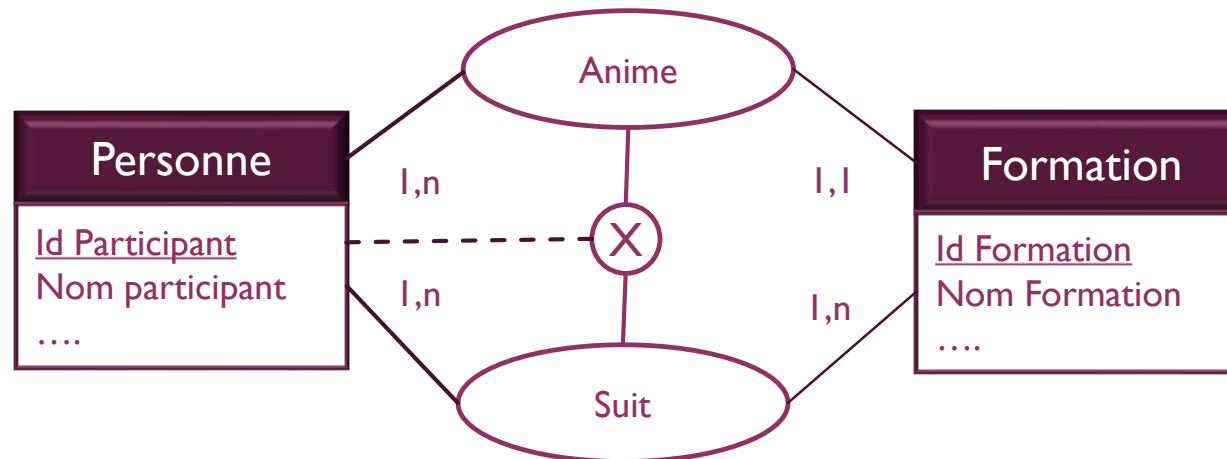
Une contrainte de totalité exprime que la totalité des occurrences d'une entité participe au moins à l'une des associations.



MÉTHODE MERISE / 2 – MCD – CONTRAINTES (3)

■ MERISE / 2 Contrainte d'exclusion

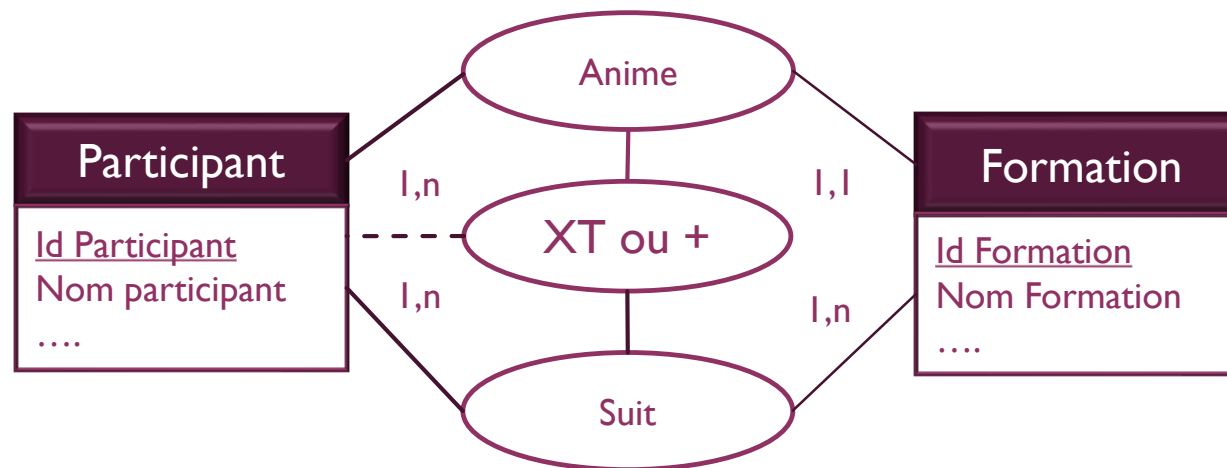
Une contrainte d'exclusion implique que pour une entité participant à deux relations, toute occurrence de l'entité participant à la première relation ne participe pas à la deuxième relation.



MÉTHODE MERISE / 2 – MCD – CONTRAINTES (4)

■ MERISE / 2 Contrainte de 'ou exclusif'

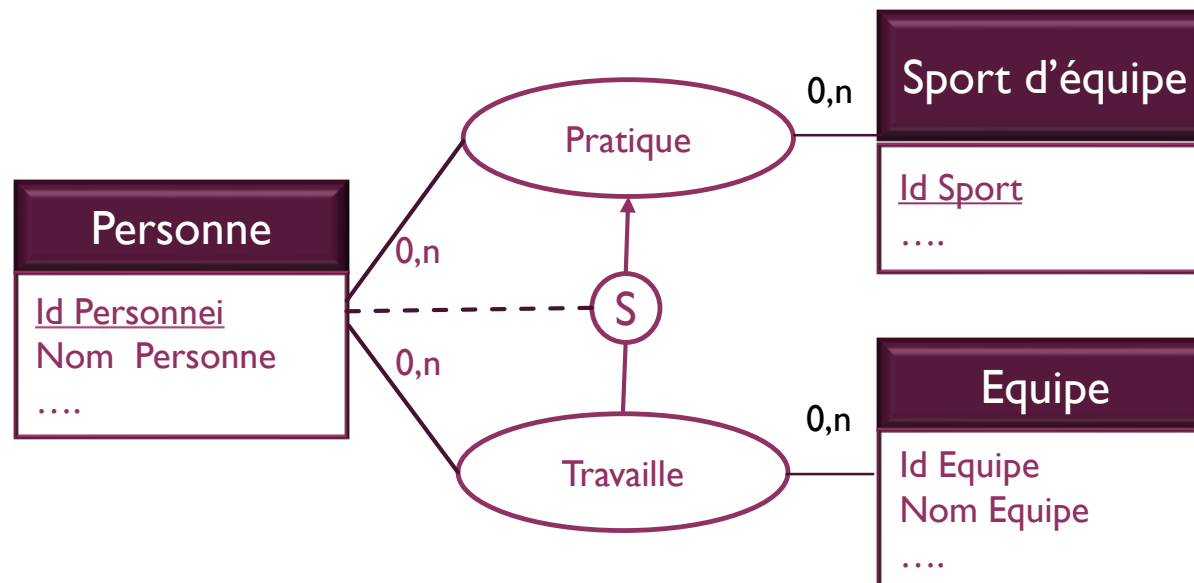
Une contrainte de 'ou exclusif' exprime la simultanéité d'une contrainte de totalité et d'exclusion



MÉTHODE MERISE / 2 – MCD – CONTRAINTES (4)

■ MERISE / 2 Contrainte d'égalité(=) ou de simultanéité (S)

Une contrainte d'égalité exprime que chaque occurrence d'une entités participant à une des relations participe à l'autre relation.



MÉTHODE MERISE – MLD – MRD (I)

■ MRD ou Modèle Relationnel de Données

- Le MRD est un Modèle logique de données appropriée à l'utilisation ultérieure d'une base de donnée relationnelle.
- Nous étudions ci-après les règles de transformation applicables au MCD pour passer d'une entité à une table et d'une relation merise également à une table. Dans un but de simplification du discours on parlera ici de table en lieu et place de 'relvar' cher aux experts du relationnel.
- Les tables relationnelle sont composées de colonnes (attributs) et de lignes (tuples)

MÉTHODE MERISE – MLD – MRD (2)

■ Traduction des Entités

- Une entité E du MCD se traduit en une table relationnelle R .
- Les propriétés de E deviennent les attributs de R .
- L'identifiant de E devient la clé primaire de R .
- Dans le cas où E est une entité de spécialisation, R hérite de la clé primaire de la table parent R_p elle-même dérivé de l'entité parent E_p .

MÉTHODE MERISE – MLD – MRD (2)

■ Traduction des Associations authentiques (pas de cardinalité 1,1)

Chaque association authentique A se traduit par une table relationnelle R dont :

- L'identifiant de chaque entité connectée à A devient un attribut de R,
- Les propriétés portées par A deviennent des attributs de R,
- La clé primaire de R est obtenue en fonction des cardinalités de A :
 - Si toutes les cardinalités sont à cardinalités maximales n, la clé primaires est la concaténation des identifiants des entités associées,
 - S'il existe au moins une cardinalité maximale à 1, il suffit de choisir une des clés candidates, chaque clé candidate correspondant à l'identifiant d'une des entités connectées à l'association par une patte à cardinalité maximale 1

MÉTHODE MERISE – MLD – MRD (3)



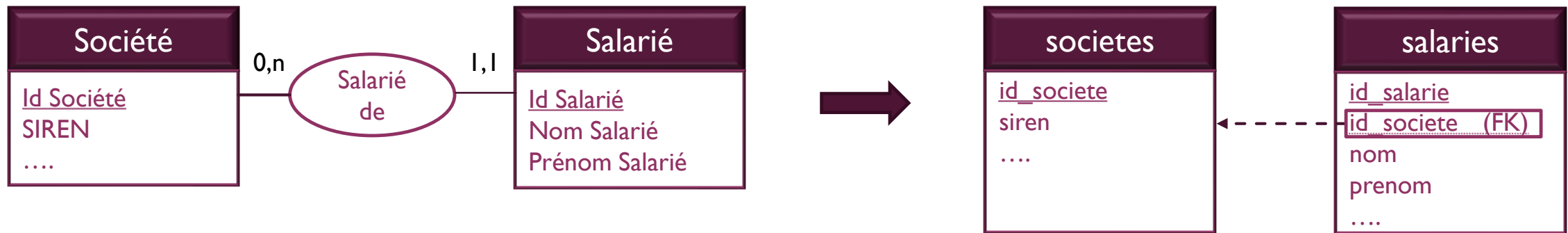
■ Associations à cardinalité 1,1 – CIF (1/2)

Dans ce cas, certains préconisent d'enrichir la table traduisant l'entité reliée par une pte 1,1 d'un attribut clé étrangère servant à référencer les autres entités participantes.

Mais attention, outre que d'un point de vue opératoire on accentue en mise à jour la pose systématique de verrous, avec les conséquences qui en découlent sur le débit transactionnel, les fameuses CIF (Contraintes d'intégrité Fonctionnelle) sont des règles qui ne sont généralement pas inscrites dans le marbre et qui peuvent changer !

MÉTHODE MERISE – MLD – MRD (3)

■ Associations à cardinalité 1,1 – CIF (2/2)



- On suppose qu'une personne n'est salariée que d'une seule société au sein d'un groupe
 - ⇒ Que se passe-t-il si la règle change et qu'un jour une personne peut être salariée par plusieurs sociétés du même groupe ?
 - ⇒ **Ce genre de 'simplification' doit être évalué au cas par cas en mesurant toutes les conséquences d'un changement de règle de gestion**

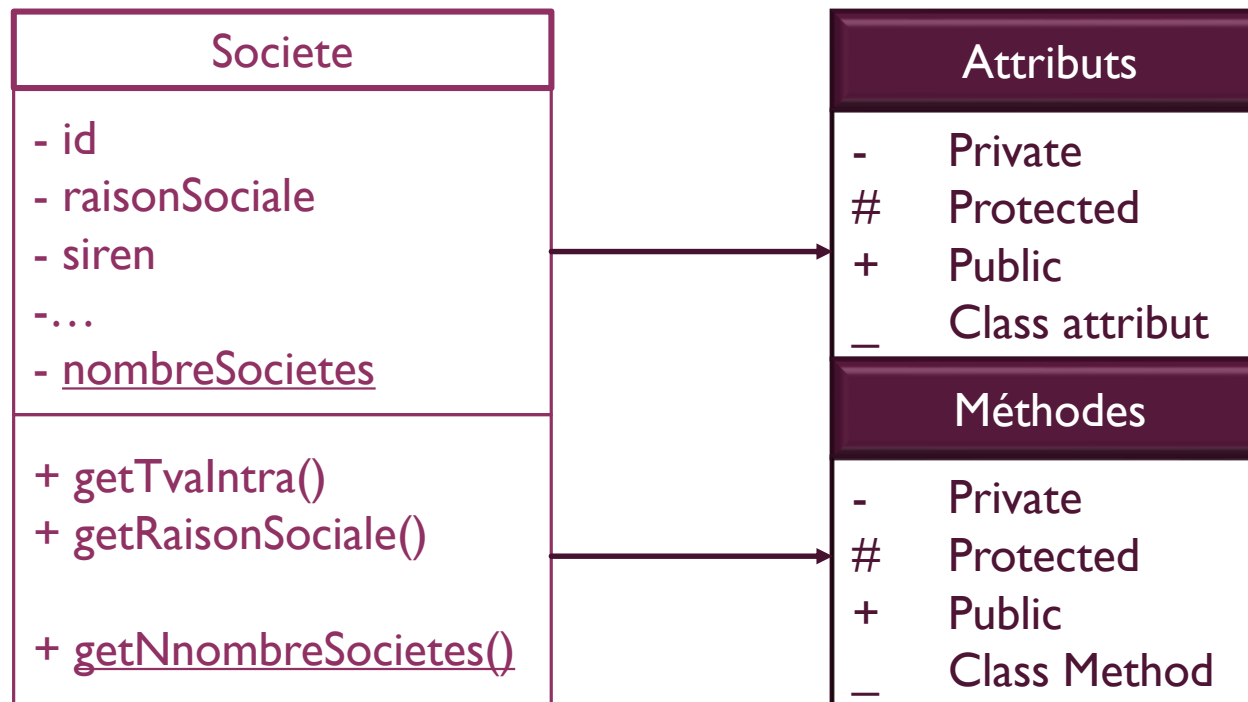


DU COTÉ D'UML


Eléments de Diagrammes de classes

UML : DIAGRAMME DE CLASSE

■ Attributs et méthodes de classe

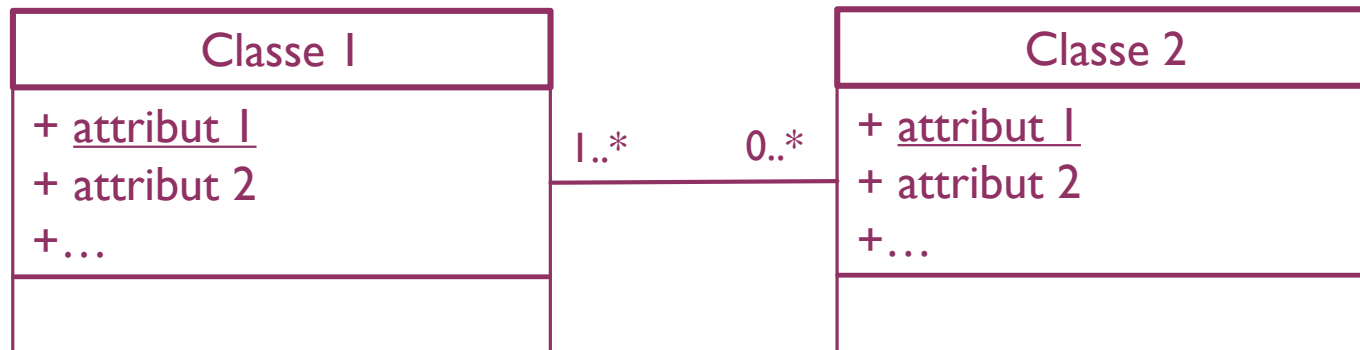


UML : DIAGRAMME DE CLASSE VS ENTITÉ MERISE

- Nom de la classe = Nom Entité Merise
- Attribut de la classe = Nom propriété Merise
 - La visibilité de l'attribut est publique
 -  Pas d'attribut de classe mais l'attribut correspondant à l'identifiant Merise est souligné
- Pas de méthode de classe

UML : GESTION DES ASSOCIATIONS MERISE SIMPLES

- L'association est matérialisée par un simple trait sur lequel on précise les cardinalités



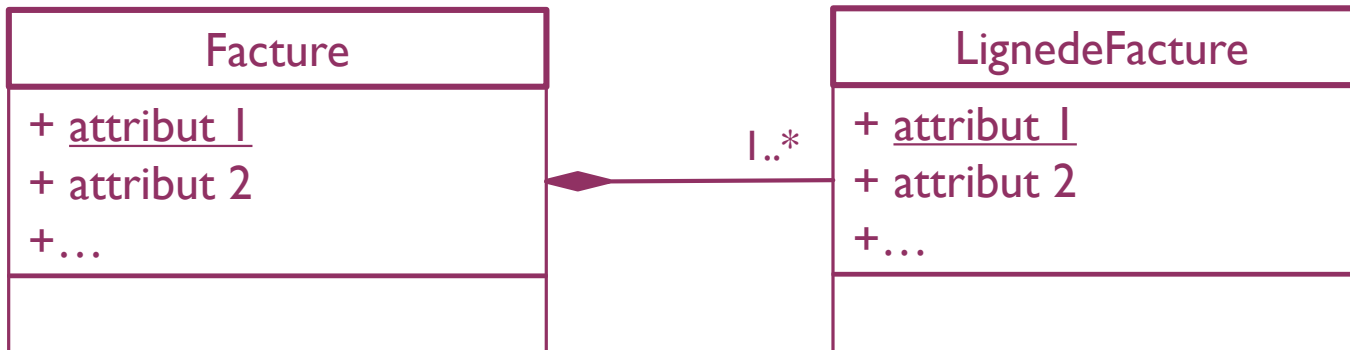
Merise	UML
0,1	0..1
1,1	1
0,n	0..* ou *
1,n	1..*

- La position des cardinalités est inversée par rapport à Merise

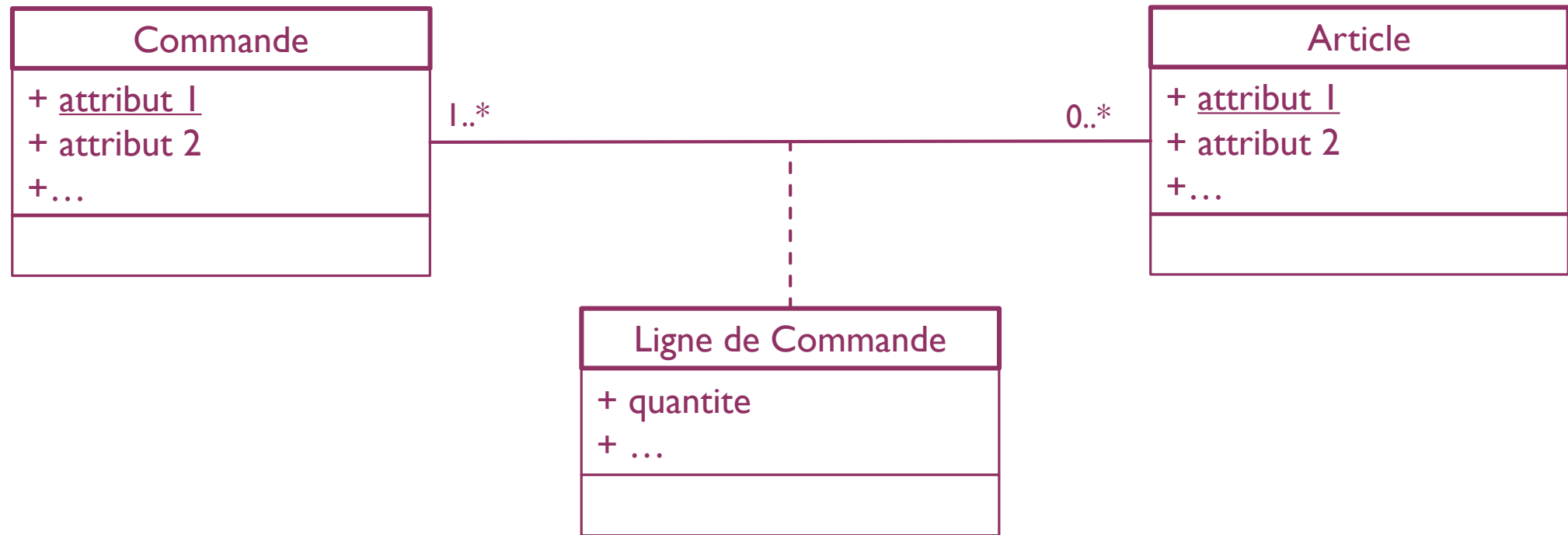


UML : PROPRIETES MULTI-VALUEES

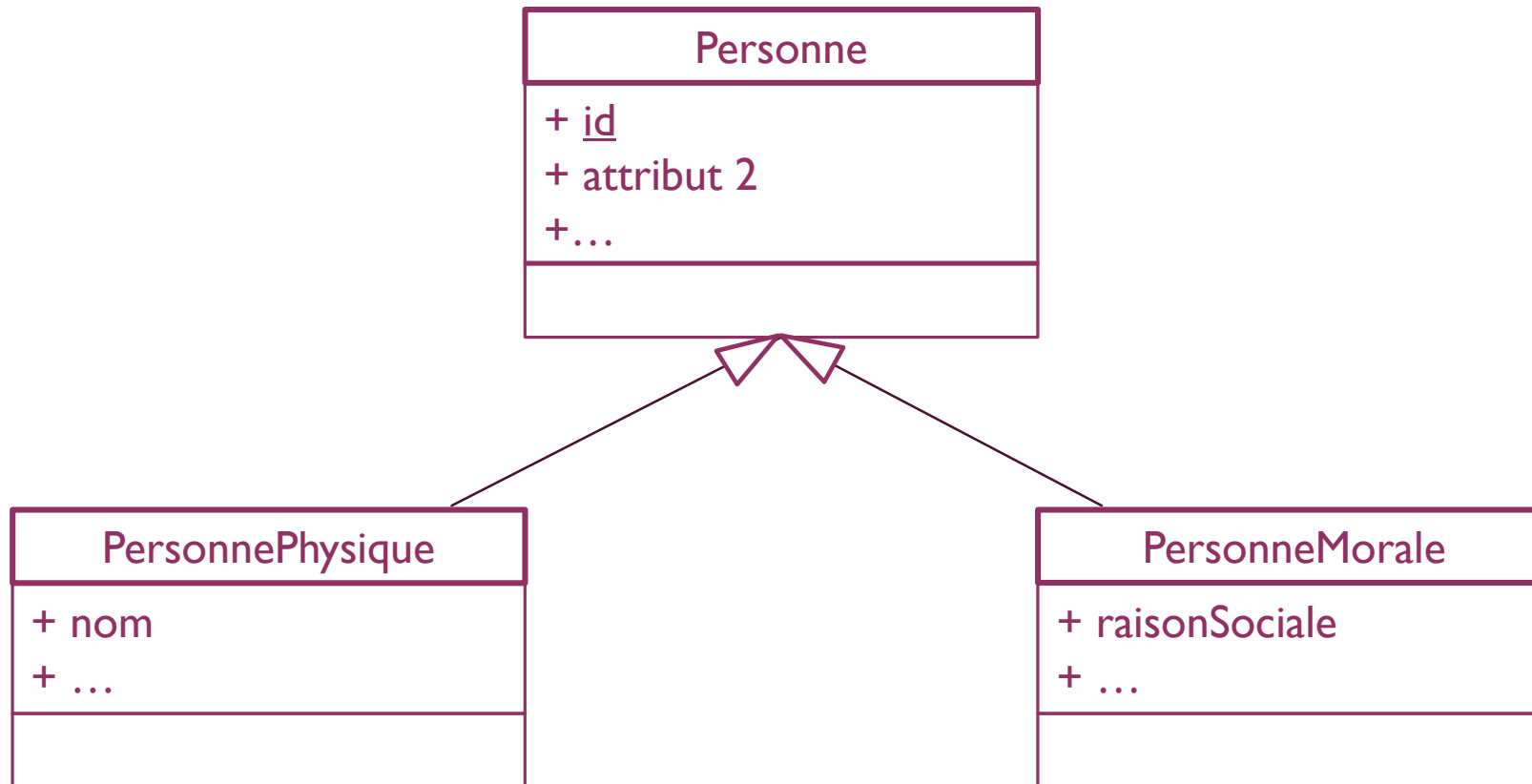
- UML dispose d'un connecteur spécifique pour la composition



UML : GESTION DES ASSOCIATIONS MERISE PORTEUSES



UML : HÉRITAGE





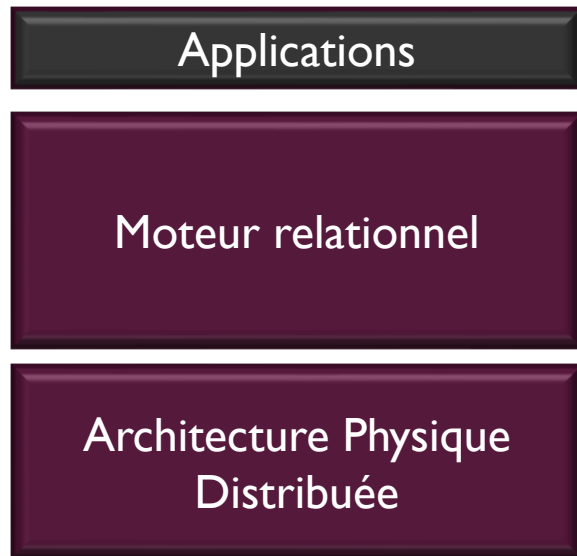
SGBDR

Quelques rappels...

INTRODUCTION

- Une base de données relationnelle :
 - Est une collection de données organisées formellement sous la forme de relations, le terme relation devant être entendu au sens mathématique de la théorie des relations elle-même construite à partir de la théorie des ensembles.
 - S'appuie sur un moteur relationnel, capable de manipuler des relations, c'est-à-dire des ensembles de données. En d'autres termes, dans le monde du relationnel on ne manipule pas des éléments mais des ensembles ! Le moteur relationnel traite les questions qu'on lui formalise sous la forme d'opérations d'algèbre relationnelle. Plus simplement, un moteur relationnel sait répondre à la question suivante : « Parmi l'ensemble des relations composant ma base de données, trouve moi l'ensemble (relation) qui a les caractéristiques suivantes ».
 - En corolaire, en relationnel on ne manipule pas des 'enregistrements' mais des ensembles entiers.
 - Modèle relationnel inventé par Edgar Frank « Ted » Codd en 1970 et construit et théorisé jusqu'au-delà des années 2000 (6NF en 2003)....

ELÉMENTS D'ARCHITECTURE



- Réalise l'indépendance entre le modèle logique (relationnel) et l'implantation physique (serveur, grappes, disques, fichier, organisation logique à l'intérieur du fichier, découpage physique du fichier, gestion index,...),
- Assure l'intégrité des données :
 - Gestion de clés étrangères,
 - Intégrité de domaine,
 - Gestion de transactions,
 - Unicité
 -
- Interface d'accès via un langage de requêtes intégrant l'algèbre relationnelle (ex SQL...)
- Optimisation des requêtes.

BASES DE DONNÉES RELATIONNELLES.....

- Alors oui, les SGBDr s'occupent du stockage des données, MAIS PAS QUE !...
- Quelques SGBDr :
 - Oracle RDBMS,
 - IBM DB2,
 - Microsoft SQL Server,
 - Oracle MySQL,
 - Maria DB,
 - PostgreSQL

COLLECTION DE DONNÉES ORGANISÉE EN TABLES

- L'organisation des tables (*) doit répondre à des règles de normalisation que l'on appelle les formes normales

- Il existe 6 niveaux de Formes Normales 1NF, 2NF,...6NF plus une forme normale, dite de BOYCE-CODD, qui intègre 2NF et 3NF et rend quelque peu caduque ces 2 dernières.
- La bible française sur le sujet est celle de François de Sainte Marie que l'on trouve ici :

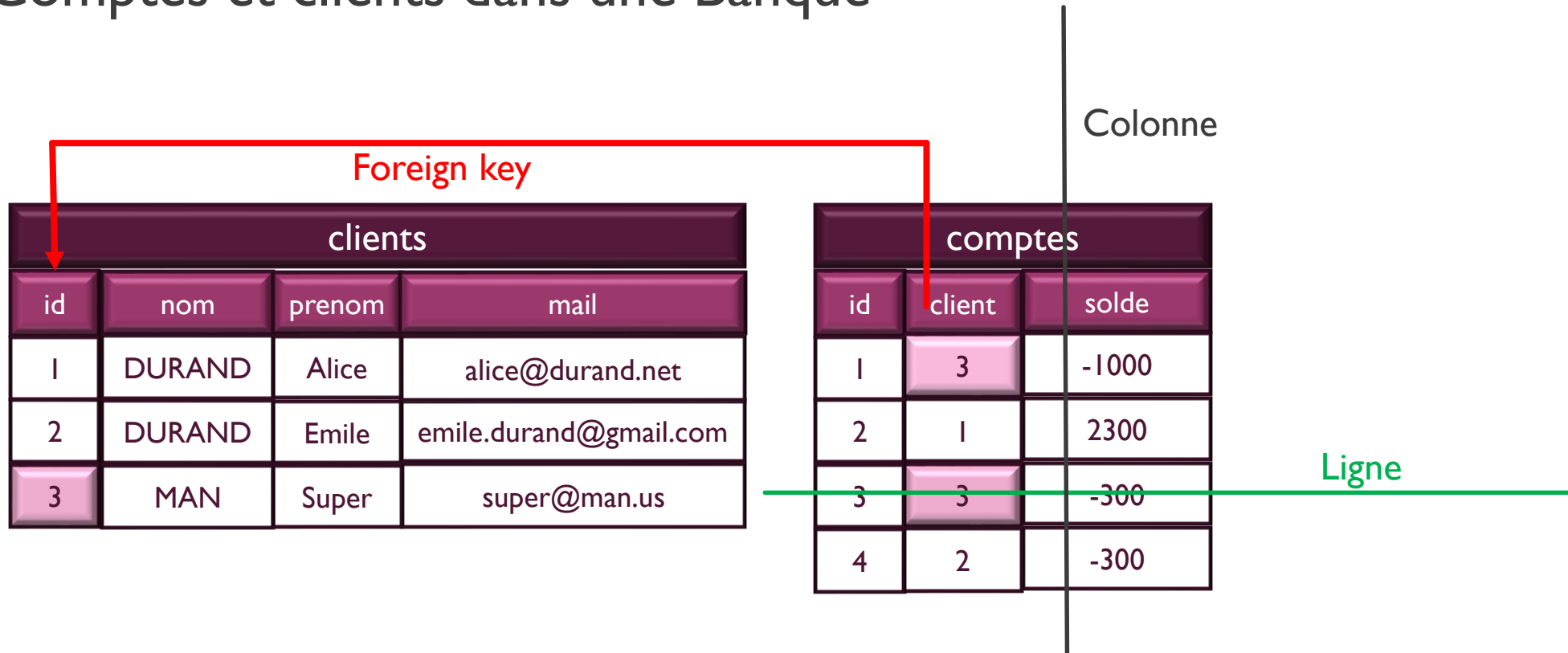
<https://fsmrel.developpez.com/basesrelationnelles/normalisation>

- Existe en version PDF (En bas à droite de la première page)
- Relisez la bible une fois, deux fois, dix fois, il restera toujours quelque chose de nouveau à comprendre !!!

(*) on verra que l'on utilisera plutôt le terme de relvar...

EXEMPLES DE TABLES 'EN RELATIONS' DANS UN SGBDR

■ Comptes et clients dans une Banque



SQL : LANGUAGE STANDARD DES SGBDR

- SQL - DDL (SQL - Data Definition Language)
 - Définition des tables
 - CREATE TABLE, DROP TABLE, ALTER TABLE
- SQL – DML (SQL – Data Manipulation Language)
 - Comme son nom l'indique : langage de manipulation de données
 - INSERT, UPDATE, DELETE, SELECT, START TRANSACTION/BEGIN, COMMIT, ROLLBACK, SAVEPOINT, ROLLBACK TO SAVEPOINT,...
- SQL – Stored Programs
 - Functions, Procedures, Triggers, Event Scheduler...

DML – UNE APPROCHE ENSEMBLISTE

■ DELETE – Exemple Syntaxe simplifiée

DELETE FROM it-akademy.clients
WHERE nom = 'DURAND' ;

Requête **ENSEMBLISTE** !



clients			
id	nom	prenom	mail
3	MAN	Super	super@man.us
4	MOUSE	Mickey	mickey.mouse@disney.com

comptes		
id	client	solde
1	3	-1000.00
3	3	-300.00
5	4	20000000.00

Suppression des
comptes
associés du fait
de la foreign key

TRANSACTIONS AND LOCKING STATEMENTS (I)

- Positionnement du problème – Cas d'un virement bancaire

compte Mickey MOUSE		
id	client	solde
5	4	20000000.00

Virement + 2000



Compte Super MAN		
id	client	solde
1	3	-1000

compte Mickey MOUSE		
id	client	solde
5	4	19998000.00

Les 2 maj ou rien



Compte Super MAN		
id	client	solde
1	3	1000

TRANSACTIONS AND LOCKING STATEMENTS (2)

■ Propriétés ACID d'une transaction (I)

- **ACID** comme **A**tomicité, **C**ohérence, **I**solation, **D**urabilité

- **Atomicité**

Assure qu'une transaction se fait au complet ou pas du tout : si une partie d'une transaction ne peut être faite, il faut effacer toute trace de la transaction et remettre les données dans l'état où elles étaient avant la transaction. L'atomicité doit être respectée dans toutes situations, comme une panne d'électricité, une défaillance de l'ordinateur.

- **Cohérence**

Assure que chaque transaction amènera le système d'un état valide à un autre état valide. Tout changement à la base de données doit être valide selon toutes les règles définies, incluant mais non limitées aux contraintes d'intégrité, aux rollbacks en cascade, aux déclencheurs de base de données, et à toutes combinaisons d'événements.

TRANSACTIONS AND LOCKING STATEMENTS (3)

■ Propriétés ACID d'une transaction (2)

■ Isolation

Toute transaction doit s'exécuter comme si elle était la seule sur le système. Aucune dépendance possible entre les transactions. La propriété d'isolation assure que l'exécution simultanée de transactions produit le même état que celui qui serait obtenu par l'exécution en série des transactions. Chaque transaction doit s'exécuter en isolation totale : si T1 et T2 s'exécutent simultanément, alors chacune doit demeurer indépendante de l'autre.

■ Durabilité

Assure que lorsqu'une transaction a été confirmée, elle demeure enregistrée même à la suite d'une panne d'électricité, d'une panne de l'ordinateur ou d'un autre problème. Par exemple, dans une base de données relationnelle, lorsqu'un groupe d'énoncés SQL a été exécuté, les résultats doivent être enregistrés de façon permanente, même dans le cas d'une panne immédiatement après l'exécution des énoncés.

TRANSACTIONS AND LOCKING STATEMENTS (4)

■ Transaction – Syntaxe (cas général)

```
START TRANSACTION;  
SELECT .....  
FOR UPDATE..... ;
```

```
UPDATE....  
UPDATE...
```

```
INSERT.....
```

```
COMMIT ; /* En cas de succès */  
ROLLBACK ; /* En cas d'anomalie */
```

Sans déclaration explicite de Transaction, le SGBD se place en autocommit. C'est une euphémisme qui signifie que toutes les mises à jour, insertions, suppressions **sont validées une par une.**

En d'autres termes, en cas d'erreur, **on ne peut plus revenir** sur les modifications de données préalablement réalisées.

} Déverrouille les données

TYPES DE DONNÉES ÉLÉMENTAIRES (I)

■ ENTIERS

- tinyint (8 bits), smallint (16 bits), mediumint (24 bits), int (32 bits), bigint (64 bits)
- (unsigned)

■ BIT(M)

■ BOOL (BOOLEAN)

- Synonyme de tinyint(1), 0 = FALSE

■ DECIMAL(M,D)

- Nombre decimal avec précision fixe (M digits, D décimales)

TYPES DE DONNÉES ÉLÉMENTAIRES (2)

■ VIRGULE FLOTTANTE

- **float** (précision ≈ 7 décimales), **double** (précision 15 \approx décimales)
- **(unsigned)**

■ Type DATE

- DATE
- DATETIME
- TIMESTAMP
- TIME
- YEAR

TYPES DE DONNÉES ÉLÉMENTAIRES (3)

- Types String
 - CHAR, VARCHAR, BINARY, VARBINARY
 - TEXT (de tiny à long...)
 - BLOB (Binary String, de tiny à long...)
 - ENUM



NORMALISATION BDR

Bases de Données Relationnelles et Formes Normales

RELATIONNEL : DÉFINITIONS (1/3)

■ Domaine (ou type)

- Rappel : ensemble de valeurs **sémantiquement cohérentes** que l'on peut vouloir on non comparer entre elles. On parle également de 'type' en lieu et place de domaine

■ n-uplet (ou tuple)

- C'est une valeur qui est un ensemble de triplets de (A_i, D_i, V_i) avec :
 - A_i : nom d'attribut,
 - D_i : nom de domaine,
 - V_i : Valeur d'attribut
- (A_i, D_i) est le $i^{\text{ème}}$ attribut, V_i en est sa valeur.
 - Exemple de n-uplet :
 $\{(Societe, entier, 1), (RaisonSociale, Char, 'IT-Akademy'), (Siren, Char, '503 253 379')\}$

RELATIONNEL : DÉFINITIONS (2/3)

■ Relation

- **Valeur** constituée d'un en-tête (ou schéma, intension) et d'un corps (ou extension)
 - L'en-tête est un ensemble de n attributs
 - Le corps est l'ensemble de n-uplets composant la relation

⇒ A la différence d'une table dans un SGBDR **une relation est invariable**
- Représentation tabulaire de la relation (Exemple)

Societe : entier	RaisonSociale : Char	Siren : Char
1	IT-Akademy	503 253 379
2	Darty	494 004 823
3	LA POSTE	356 000 000

RELATIONNEL : DÉFINITIONS (3/3)

■ Relvar (ou Variable Relationnelle)

Une relvar est une variable affectée successivement de valeurs qui sont des relations partageant la même en-tête.

⇒ L'analogie avec une table de SGBDR se fait sur la relvar et non pas sur la relation.

RELATIONNEL : OBJECTIFS DE LA NORMALISATION

- Intégrité des données
- Minimiser les redondances
- Evolutivité du système cible
- Performances du système global
 - Dénormaliser pour optimiser les performances ?



RELATIONNEL : DÉMARCHE DE NORMALISATION

Relvars 1NF

Relvars 2NF

Relvars 3NF

Relvars BCNF

Relvars 4NF

Relvars 5NF

Relvars 6NF

- une relvar en $(n+1)$ NF implique (n) NF
- on part des tables issues du MLD
- on vérifie qu'elles respectent 1NF.
- on entame une démarche de normalisation de plus en plus poussée

RELATIONNEL : INF - PREMIÈRE FORME NORMALE (1/4)

■ Définition(s)

E.F Codd en 1971, dans « Further Normalization of the Data Base Relational Model ».

« Une relation est en première forme normale si aucun de ses domaines ne peut contenir des éléments qui soient eux-mêmes des ensembles. Une relation qui n'est pas en première forme normale est dite non normalisée ».

Traduit par des théoriciens du relationnel :

« une relation est dite « normalisée » ou en « première forme normale » (INF) si aucun attribut qui la compose n'est lui-même une relation, c'est-à-dire, si tout attribut est atomique (non décomposable). » Serge Miranda en 1986

RELATIONNEL : INF - PREMIÈRE FORME NORMALE (2/4)

■ Atomicité

On peut chercher à pinailler sur le caractère atomique d'un attribut, mais ce qui est à proscrire est le caractère répétitif des valeurs au sein même d'un attribut faisant référence à un domaine donné.

Citons C. Date en 2007 : « Une colonne (attribut) C constitue un « groupe répétitif » si étant définie sur le domaine D, les valeurs légales pouvant apparaître dans C sont des ensembles (ou des listes, des tableaux, ou ...) de valeurs du domaine D. Ces groupes répétitifs sont définitivement proscrits du Modèle Relationnel »

RELATIONNEL : INF - PREMIÈRE FORME NORMALE (3/4)

■ INF ?

Id : entier	Nom : Char	Téléphone
I	IT-Akademy	04 45 43 56; 04 45 67 32

Id : entier	Nom : Char	Tel1	Tel2
I	IT-Akademy	04 45 43 56	04 45 67 32

RELATIONNEL : INF - PREMIÈRE FORME NORMALE (4/4)

■ INF ? So What ?

- Considérons la relvar suivante :

<u>IdClient</u>	<u>NumCommande</u>	RaisonSociale	DateCommande	Etat
12	1	IT-Akademy	2018-10-08	Signée
3	1	LA POSTE	2018-10-15	Refusée
12	2	IT-Akademy	2018-11-20	Payée

- Cette relvar est en INF,
- MAIS ?

TP - I

- Etablir le MRD d'un système comprenant :
 - Des Sociétés d'un groupe
 - Propriétés : Code Societe (unique), Raison Sociale, SIREN, Statut Juridique
 - Des Clients des Sociétés du groupe
 - Propriétés : raison sociale, SIREN, Statut Juridique, contacts, téléphones contacts, emails contacts
 - Des Commerciaux des sociétés
 - le commercial peut-être pluri-sociétés
 - Le commercial gère un portefeuille de clients et est l'interlocuteur unique d'un contact client
 - Propriété : nom, prénom, email, téléphones, login, mot de passe

TP - 2

■ Modifier le MRD

- Créer les tables des personnes physiques et personnes morale
- Un client peut être une personne physique ou morale
- Créer une table rcs (registre du commerce)
 - Propriétés : raison sociale, immatriculation (Siren), immatriculation Siège (Siret), NumeroTVA, Activité, Capital
- Faire en sorte que toute personne physique puisse avoir un login/mot de passe
- Gérer la contrainte suivante : on ne peut être à la fois personne physique et personne morale (contrainte Merise/2 de 'ou exclusif')
- Facultatif : gérer les adresses des personnes physiques et morales

ÉLÉMENTS DE PROGRAMMATION DES TRIGGERS (I)

■ Types de triggers (MySQL)

- Un trigger est lié à une table
- Un trigger peut être positionné sur une opération d'insertion, d'update ou de delete
- Sur chaque type d'opération (insertion, update, delete), on peut positionner un trigger avant l'opération (trigger before) ou après (trigger after).

■ Référence des valeurs d'une colonne pour la table cible d'un trigger

- On utilise les mots clés NEW.nom_colonne ou OLD.nom_colonne pour faire référence à ces valeurs.
- OLD permet de connaître la valeur de la colonne avant opération et NEW est la valeur qui s'apprête à valoriser la colonne

ÉLÉMENTS DE PROGRAMMATION DES TRIGGERS (2)

■ Éléments de langage

- On encadre le code proprement dit du trigger par une séquence BEGIN.....END
- Les instructions se terminent par un classique ‘;’
- Les variables se déclare par une clause DECLARE :
DECLARE nomVariable Type(int, varchar,...) [DEFAULT Valeur] ;
- L’affectation se fait par l’instruction SET :
SET nomVariable = Valeur ;
- La syntaxe du ‘if’ est la suivante :
IF (Condition)
THEN /* Code à exécuter si Condition est vraie */
[ELSE /* Code à exécuter si Condition est fausse */]
END IF ;

ÉLÉMENTS DE PROGRAMMATION DES TRIGGERS (3)

■ Exemple type

BEGIN

DECLARE nomVariable INTEGER(10) UNSIGNED DEFAULT 0 ;

/*

** (T) Code du trigger

*/

/*

** (EXIT)

*/

END

RELATIONNEL : BCNF (BOYCE-CODD NF) – (1/6)

■ Énoncé

- Une relvar R est en forme normale de Boyce-Codd si et seulement si, pour chaque dépendance fonctionnelle non triviale $A \rightarrow B$ qui doit être vérifiée par R , A est une surclé de R .
- Certes !

RELATIONNEL : BCNF (BOYCE-CODD NF) – (2/6)

- Reprenons la relvar précédente :

<u>IdClient</u>	<u>NumCommande</u>	RaisonSociale	DateCommande	Etat
12	1	IT-Akademy	2018-10-08	Signée
3	1	LA POSTE	2018-10-15	Refusée
12	2	IT-Akademy	2018-11-20	Payée

- Cette relvar n'est pas en BCNF car
 - Il existe une DF non triviale : $\{\text{idClient}\} \rightarrow \{\text{RaisonSociale}\}$
 - $\{\text{idClient}\}$ n'est pas une surclé
- ⇒ il faut normaliser (en BCNF) cette relvar délinquante



RELATIONNEL : BCNF (BOYCE-CODD NF) – (3/6)

■ Théorème de Heath (1/2)

Soit la relvar $\{A, B, C\}$ dans laquelle A , B , et C sont des sous-ensembles d'attributs de R .

Si R satisfait à la DF $\{A\} \rightarrow \{B\}$ alors R est égale à la jointure de ses projections sur $\{A, B\}$ et $\{A, C\}$.

Donc : si $R1 = \{A, B\}$ et $R2 = \{A, C\}$, alors $R = R1 \text{ JOIN } R2$

⇒ Permet donc de décomposer une relvar non normalisée pour normaliser



Attention de ne pas tolérer les NULL !

RELATIONNEL : BCNF (BOYCE-CODD NF) – (4/6)

■ Théorème de Heath (2/2)

- Préservation des dépendances

Lors de la décomposition par projection/jointure, il faut veiller à ne pas perdre des DF et appliquer la règle de Jorma Rissanen :

Soit la relvar $\{A, B, C\}$ dans laquelle A , B , et C sont des sous-ensembles d'attributs de R .

Si R satisfait à la DF $\{A\} \rightarrow \{B\}$ et $\{B\} \rightarrow \{C\}$ alors on décomposera R selon la jointure de ses projections sur $\{A, B\}$ et $\{B, C\}$.

RELATIONNEL : BCNF (BOYCE-CODD NF) – (5/6)

- Normalisation une relvar R non BCNF (1/2)
 - On décompose :
 - par projection sur la DF cause du viol de BCNF, on décompose R en 2 relvars R1 et R2 'mieux' normalisés tels que $R = R1 \text{ JOIN } R2$
 - On recommence si nécessaire tant qu'il est nécessaire de normaliser

RELATIONNEL : BCNF (BOYCE-CODD NF) – (6/6)

■ Normalisation une relvar R non BCNF (2/2)

- La décomposition de la relvar 'commande' précédemment vue se fait donc ainsi : $R = R1 \text{ JOIN } R2$

<u>IdClient</u>	RaisonSociale
3	LA POSTE
12	IT-Akademy

R1

<u>IdClient</u>	<u>NumCommande</u>	DateCommande	Etat
12	1	2018-10-08	Signée
3	1	2018-10-15	Refusée
12	2	2018-11-20	Payée

R2

TP - 3

- Vérifier que la solution mise en place dans TP-2 est bien en BCNF

RELATIONNEL : 2NF - SECONDE FORME NORMALE (1/2)

■ Énoncé

Un relvar R est en deuxième forme normale si :

- Elle est en 1^{ère} forme normale,
- Chaque attribut n'appartenant à aucune clé candidate de R est en dépendance totale de chaque clé candidate de R



Attention : on trouve souvent l'énoncé erroné suivant : « Une relation R est en deuxième forme normale si elle est en 1^{ère} forme normale et si chaque attribut n'appartenant pas à la clé primaire de R est en dépendance totale de celle-ci. »

RELATIONNEL : 2NF - SECONDE FORME NORMALE (2/2)

- Ajoutons une clé primaire absolue à la relvar précédente :

<u>Id Commande</u>	IdClient	NumCommande	RaisonSociale	DateCommande	Etat
1	12	1	IT-Akademy	2018-10-08	Signée
2	3	1	LA POSTE	2018-10-15	Refusée
3	12	2	IT-Akademy	2018-11-20	Payée

- Cette relvar n'est pas en NF2
 - Car {RaisonSociale} est en dépendance partielle non totale de la Clé candidate {idClient, NumCommande}
- Elle serait en 2NF selon la définition erronée de la 2NF !!!

RELATIONNEL : 3NF - TROISIÈME FORME NORMALE (I)

■ Énoncé

Un relvar R est en troisième forme normale si :

- Elle est en deuxième forme normale,
- Chaque attribut n'appartenant à aucune clé candidate de R ne dépend que directement des clés candidates de R

RELATIONNEL : 3NF - TROISIÈME FORME NORMALE (2)

■ Etude de la relvar précédente

Si on met 'de côté' le fait que cette relvar n'est pas 2NF, on compte les DF suivantes :

- (DF1) : {Id Commande} → {IdClient}
- (DF2) : {Id Commande} → {NumCommande}
- (DF3) : {Id Commande} → {RaisonSociale}
- (DF4) : {Id Commande} → {DateCommande}
- (DF5) : {Id Commande} → {Etat}

- (DF6) : {IdClient} → {RaisonSociale}

- On retrouve (DF3) par transitivité (cf. Armstrong) par transitivité de (DF1) et (DF2)
⇒ {RaisonSociale}, dépend transitivement de {Id Commande} via {IdClient} qui n'est pas dans la clé

RELATIONNEL : BCNF LA PANACÉE ? (1/3)

- La décomposition par projection/jointure peut perdre des DF

Exemple de C. DATE (2004) : relvar EMP

<u>Etudiant</u>	Matiere	Professeur
Jules	Maths	DURAND
Jules	Anglais	SMITH
César	Maths	DURAND
César	Anglais	DUBOIS

(G1) : Pour chaque matière étudiée, un étudiant n'a qu'un seul professeur

(G2) : Chaque professeur n'enseigne qu'une matière

(G3) : Chaque matière peut être enseignée par plusieurs professeurs

(G4) : Chaque professeur peut enseigner à plusieurs étudiants

RELATIONNEL : BCNF LA PANACÉE ? (2/3)

■ En matière de DF :

(DF1) : {Etudiant, Matiere} \rightarrow {Professeur}

(DF2) : {Professeur} \rightarrow {Matiere}

On applique le théorème de Heath sur la base de (DF2) en décomposant la relvar EMP

EMP = PM JOIN PE avec :

PM : {Professeur, Matiere} clé candidate unique : {Professeur}

PE : {Professeur, Etudiant} clé candidate unique : {Professeur, Etudiant}

RELATIONNEL : BCNF LA PANACÉE ? (3/3)

- On obtient :

PM

<u>Professeur</u>	Matière
DUBOIS	Anglais
DURAND	Maths
SMITH	Anglais

PE

<u>Professeur</u>	<u>Etudiant</u>
DUBOIS	César
DURAND	César
DURAND	Jules
SMITH	Jules

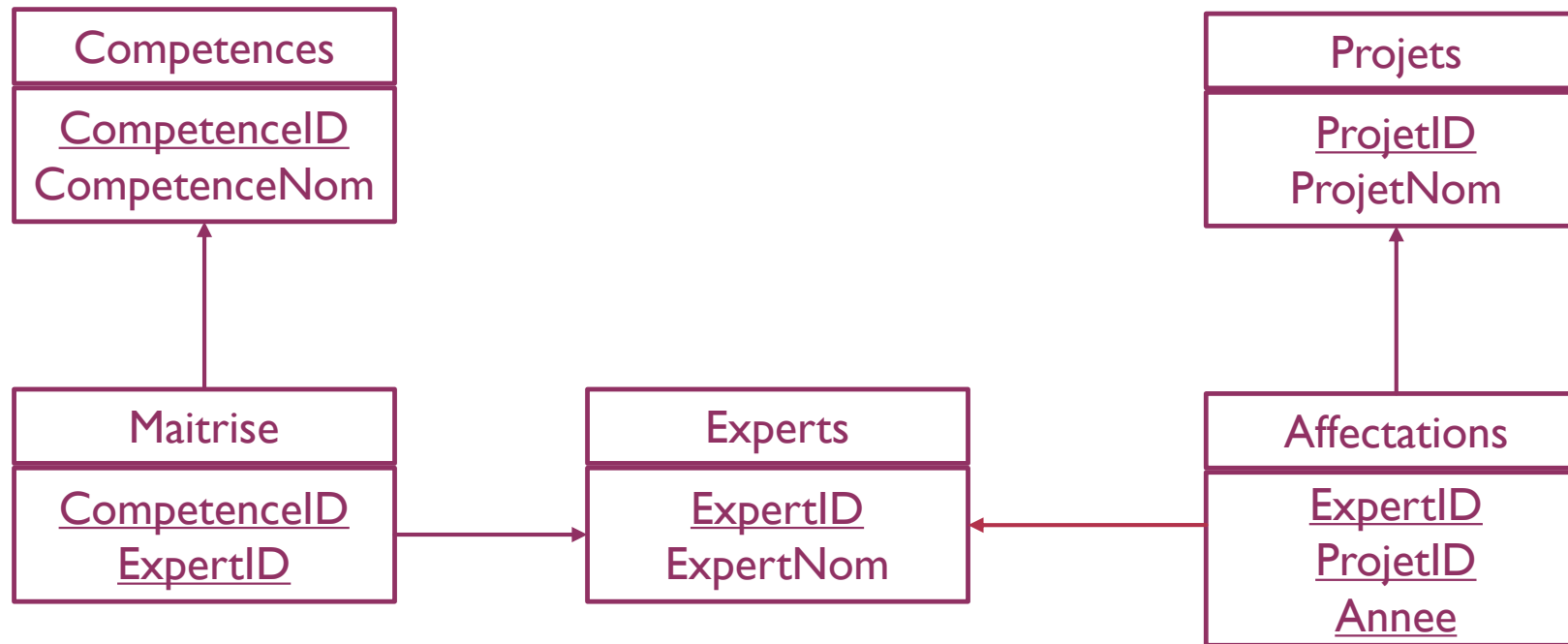
- Les 2 relvars sont en BCNF
- EMP = PM JOIN PE
- **Mais on perd DFI** (règle de gestion GI) ! Et Il faudra gérer GI par un trigger.....

TP - 4

- A partir du modèle TP-2:
 - Comment peut-on trouver la liste des Clients d'une Société ?

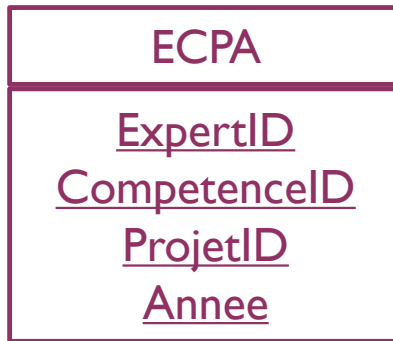
RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (I/I0)

■ Modélisation activité SSII



RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (2/10)

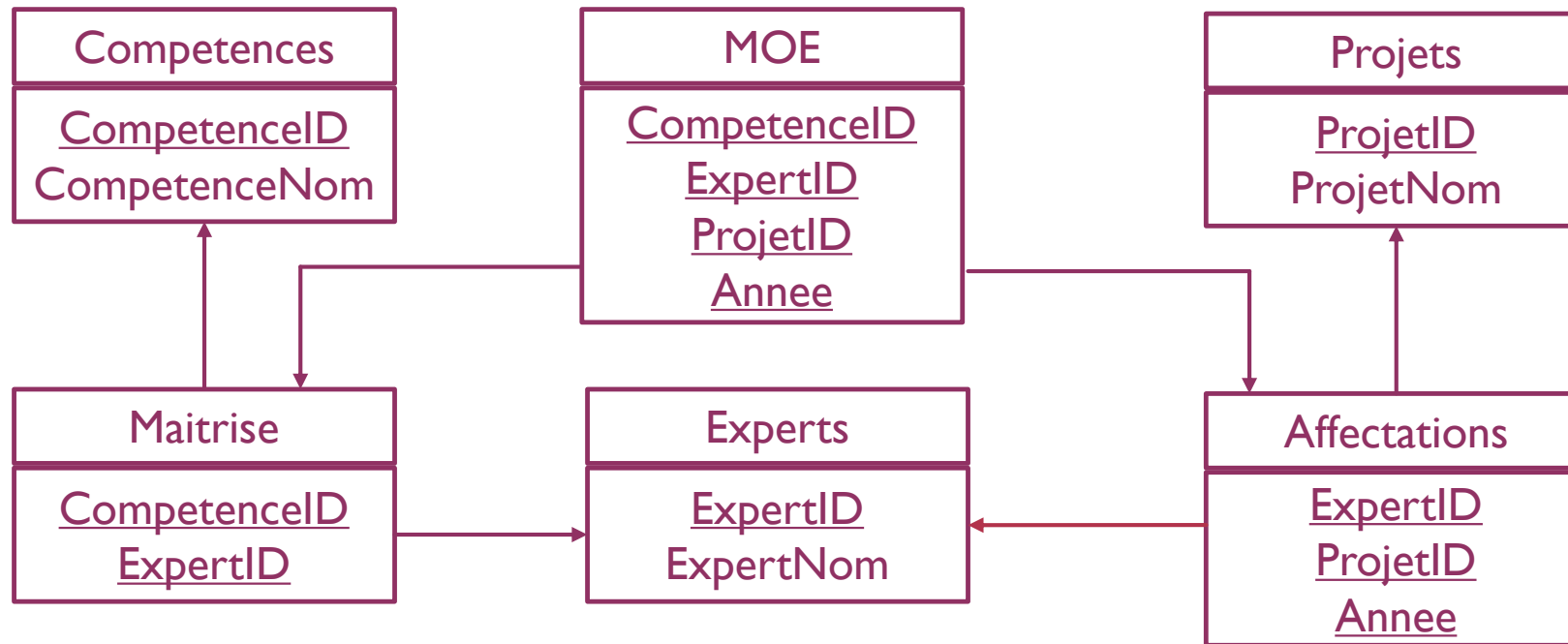
- Soit ECPA = Maitrises JOIN Affectations



- ECPA est en BCNF (Toutes les dépendances sont triviales)
- ECPA est ultra redondante 😬
- Piège fatal ⚠ ECPA porte l'information suivante :
 - ExpertID, qui a la compétence CompetenceID a été affecté sur ProjetID l'année Annee
 - **Et surtout pas** : ExpertID a mis en œuvre la compétence Competence sur ProjetID l'année Annee (Prédicat (P))

RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (3/10)

- Pour que le prédicat (P) soit vrai



RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (4/10)

■ Dépendances multivaluées :

Soit une relvar R $\{X, Y, Z\}$ X, Y , et Z étant des sous-ensembles de l'entête H de R avec $Z = H - X - Y$

Si pour chaque valeur prise par X , les valeurs prises par Y sont indépendantes des valeurs prises par Z alors il existe une dépendance multivaluée $X \twoheadrightarrow Y$ (X et Y non nécessairement disjoints)

Dans le cas de ECPA :

$\{\text{ExpertID}\} \twoheadrightarrow \{\text{CompetenceID}\}$

$\{\text{ExpertID}\} \twoheadrightarrow \{\text{ProjetID}, \text{Annee}\}$

En clair : les compétences n'ont rien à voir avec les affectations !

RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (5/10)

■ Théorème de Fagin

Soit une relvar R $\{X, Y, Z\}$ X, Y , et Z étant des sous-ensembles d'attributs. R est égale à la jointure de ses projections sur $\{X, Y\}$ et $\{X, Z\}$ si et seulement si R vérifie les dépendances multivaluées $X \twoheadrightarrow Y$ et $X \twoheadrightarrow Z$.

Y et Z étant symétrique on en déduit la règle de complémentation : la dépendance multivaluée $X \twoheadrightarrow Y$ est vérifiée si et seulement si $X \twoheadrightarrow Z$.

On écrit de manière plus concise $X \twoheadrightarrow Y \mid Z$

RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (6/10)

■ Et le théorème de Heath ?

Par application du théorème de Heath :

Soit une relvar R $\{X, Y, Z\}$ X, Y , et Z étant des sous-ensembles d'attributs.

Si $X \rightarrow Y$ alors $R = \{X, Y\} \text{ JOIN } \{X, Z\}$.

Le théorème de Fagin nous permet d'en déduire que $X \twoheadrightarrow Y$ (et $X \twoheadrightarrow Z$).

On en déduit donc que $X \rightarrow Y \Rightarrow X \twoheadrightarrow Y$

RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (7/10)

■ Dépendance multivaluée triviale

Soit X et Y deux sous-ensembles d'attributs de l'entête H d'une relvar R . La dépendance multivaluée $X \twoheadrightarrow Y$ est triviale si $X \supseteq Y$ ou $H = X \cup Y$

■ Énoncé de la 4NF :

Soit R une relvar et X et Y deux sous-ensembles d'attributs quelconques de l'entête H de R . R est en 4NF si et seulement si, pour chaque dépendance multivaluée $X \twoheadrightarrow Y$ vérifiée pour R , au moins une des conditions suivantes est satisfaite :

- La dépendance $X \twoheadrightarrow Y$ est triviale,
- $X \rightarrow \{A\}$ pour chaque attribut de H .

X est alors une surclé de H

RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (8/10)

■ Cas de ECPA

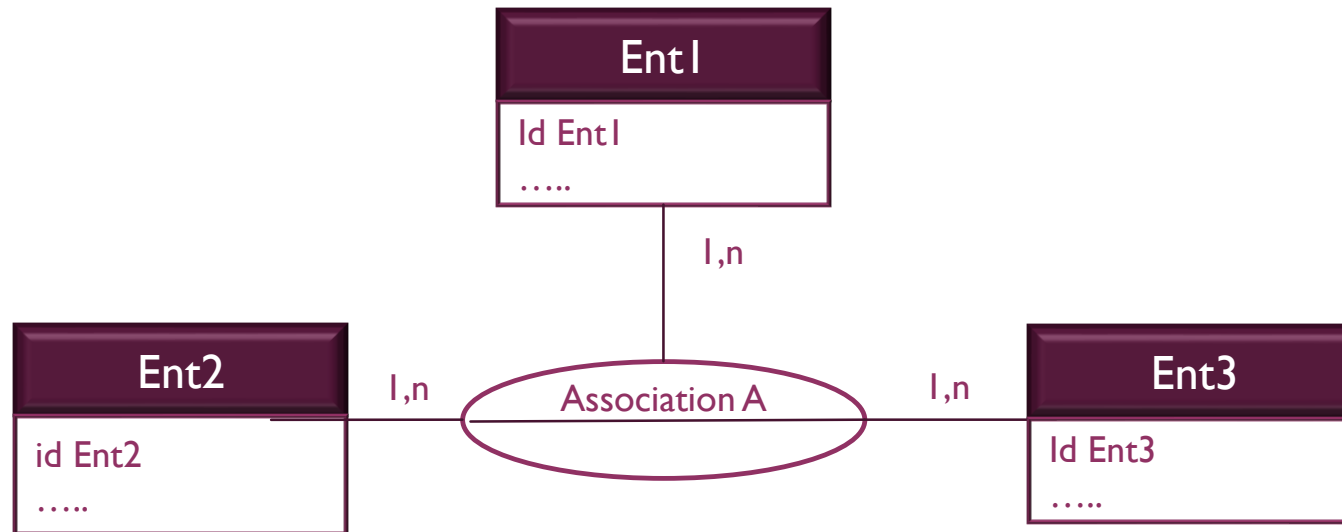
- ECPA est en BCNF (dépendance triviale)
- ECPA n'est pas en 4NF car
 - $\{\text{ExpertID}\} \rightarrow\rightarrow \{\text{CompetenceID}\}$

$\{\text{ExpertID}\}$ n'est pas une surclé de ECPA qui ne comprend comme seule surclé que $\{\text{ExpertID}, \text{CompetenceID}, \text{ProjetID}, \text{Annee}\}$

⇒ Si ECPA 'était un table candidate' de la base de données, il faudrait la décomposer en utilisant le théorème de Fagin

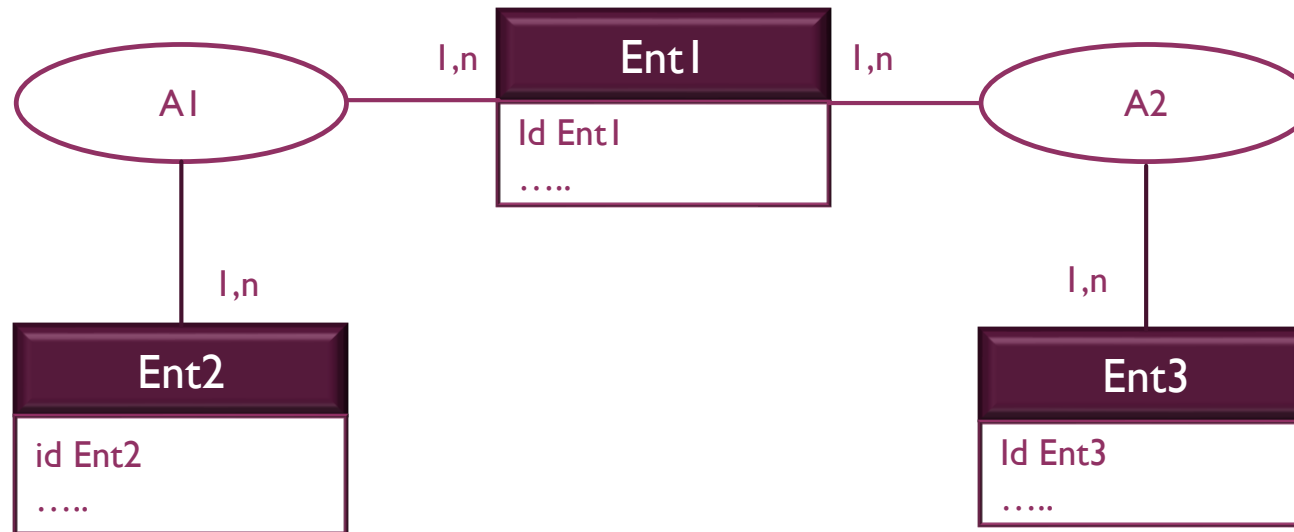
RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (9/10)

- Cas pratique de détection de viol de 4NF dans Merise (1/2)
 - Concerne les associations ternaires et plus, dont il faut se méfier !



RELATIONNEL : QUATRIÈME FORME NORMALE – 4NF (10)

- Cas pratique de détection de viol de 4NF dans Merise (2/2)
 - Qui, du fait d'une indépendance entre Ent2 et Ent3 est en fait :



TP - 5

■ Ajouter dans le modèle du TP-2 :

- Fournisseurs,
- Factures Clients et Factures Fournisseurs

On gèrera ;

- Le montant HT de la facture et la date de facture
- La date de réception pour les factures fournisseurs
- le numéro de facture pour les factures clients (incrémentation du numéro par Société) ainsi que le destinataire (contact client)
- Commissionnement des commerciaux (% commissionnement individualisé pour chaque commercial en fonction de chaque société)
- Produire la liste des clients par sociétés, le chiffre d'affaire par client pour chaque société ainsi que le montant des commissions versées à chaque commercial sur une période donnée.

ANNEXE - ROUTINES STOCKÉES (MYSQL)

- Catégories :
 - Fonctions, Procedures,
 - Triggers,
 - Event Scheduler

Pour la syntaxe déclarative et les détails du langage, on se reportera à la documentation MySQL : <https://dev.mysql.com/doc/refman/5.7/en/stored-programs-views.html> MySQL

On trouvera de larges exemples ici : <https://github.com/isabelleLetrong/zxitMySQL>

TP - 6

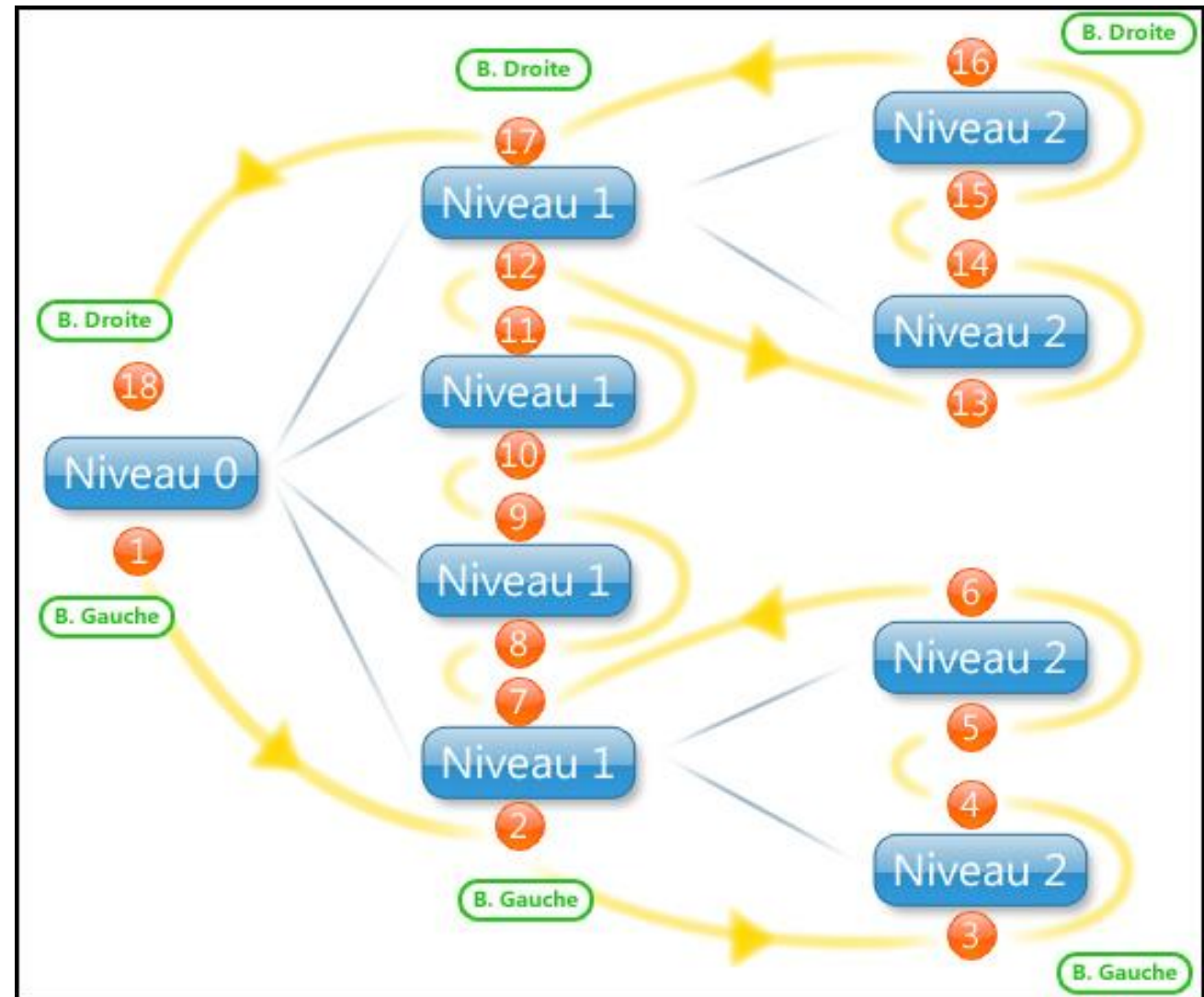
■ A partir du modèle TP-2:

Créer la procédure stockée `societes_create()`

- Paramètres en entrée : Raison Sociale, Siren, Siret, TVA Intra, Activité, Capital, Statut Juridique
- Paramètres en sortie : Status (entier), Step(entier), id

TP - 7

- Gestion de nomenclature (I)
 - La nomenclature est conçue comme un arbre géré par représentation intervallaire.
 - Pour mémoire l'arbre est conçu de la manière suivante :



TP – 7 – GESTION DE NOMENCLATURE (2)

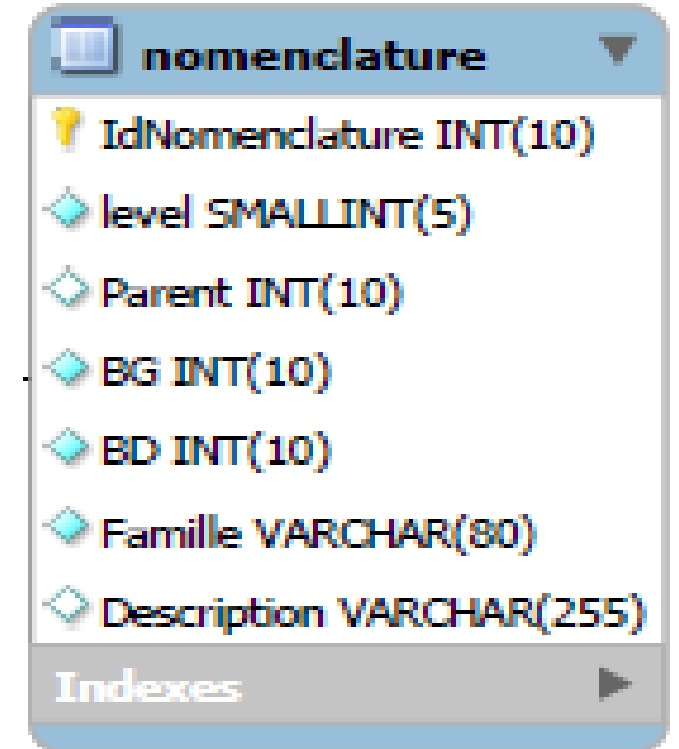
- Dans une représentation intervallaire, chaque article de la nomenclature est caractérisé par un Bord Gauche (BG dans la table nomenclature) et un Bord Droit (BD dans la table nomenclature). Ce sont ces 2 bords qui caractérisent la position de l'article dans l'arbre. On note les caractéristiques suivantes :
 - La Racine (Niveau 0) possède à l'origine (i.e. lorsque l'arbre ne contient que la racine) les valeurs $BG = 1$ et $BD = 2$,
 - Pour toute feuille (extrémité de l'arbre) $BD - BG = 1$,
 - Pour tout nœud de la nomenclature, $BD - BG - 1 = \text{nombre d'articles enfants}$.

TP – 7 – GESTION DE NOMENCLATURE (3)

■ Table Nomenclature

De manière à faciliter les recherches et la présentation des données, en complément de BG et BD, la table contient les colonnes suivantes :

- **Level** : Définit le niveau dans l'arbre de l'article,
- **Parent** : Pointe sur l'article parent,
- **Famille** : Identifiant de la Famille, la valeur de **Famille** est unique dans l'arbre
- **Description** : Précise la définition de **Famille**.



TP – 7 – GESTION DE NOMENCLATURE (4)

■ Créer la procédure stockée

```
CREATE PROCEDURE `nomenclature_insertFamily`  
(  
  OUT    $Status    INTEGER(10) UNSIGNED ,  
  OUT    $Step      INTEGER(10) UNSIGNED ,  
  OUT    $FamilyID  INTEGER(10) UNSIGNED ,  
  OUT    $Level     SMALLINT(5) UNSIGNED ,  
  
  IN     $Parent    VARCHAR(128)      ,  
  IN     $Family    VARCHAR(128)      ,  
  IN     $Description VARCHAR(255))    ,  
)
```

TP – 7 – GESTION DE NOMENCLATURE (5)

- Ecrire la requête qui effectue la recherche d'une Famille et de ses enfants