

Cours

SGBD

Pr: BELMAJDOUB Hanae

SUPMTI

belmajdoubsupmti@gmail.com

Plan

- **Chapitre 1 Introduction générale**
- **Chapitre 2 Le modèle relationnel**
- **Chapitre 3 Présentation des données**
- **Chapitre 4 L'algèbre relationnelle**
- **Chapitre 5 Le langage QBE**
- **Chapitre 6 Le langage SQL**

Chapitre 1 Introduction générale

I. Notions intuitives

I. Objectifs et avantages des SGBD

I. L'architecture ANSI/SPARC

I. Notion de modélisation des données

I. Survol des différents modèles de données

I. Bref historique, principaux SGBD commercialisés

I: Notions Intuitives

Base de données

Une base de données est un ensemble de données stockées dans un système informatique telles qu'elles sont :

structurées ; organisées pour répondre rapidement à des questions précises utilisent un SGBD (Système de Gestion de Bases de Données) = logiciel de consultation et de mise à jour.

Une BD est faite pour enregistrer des faits, des opérations au sein d'un organisme (administration, banque, université, hôpital, ...)

Les BD ont une place essentielle dans l'informatique

I: Notions Intuitives

Base de données

Une banque de données est une base de données telle que les données sont : relatives à un domaine défini de connaissance facilement consultable par un grand nombre d'utilisateurs

Les bases de données sont utilisées dans différents domaines d'application. On parle alors de BD :

- factuelles : administration, gestion, données structurées ;
- documentaires : bibliothèque, peu évolutives, peu de suppression ;
- cartographiques : CAO, figures géométriques ; images : matrices de points ;
- textuelles : textes libres, peu structurées ; informatiques : AGL, objets informatiques ;
- multimédia : factuelles + textuelles + images + son = hypertexte.

I: Notions Intuitives

Base de données

Une base de données est la représentation de la réalité sous forme de données interreliées telles qu'elles soient :

- Opérationnelles
- Enregistrées
- Utilisées par des systèmes d'application

Il faut donc minimiser la redondance et structurer les données de façon à supporter les accès multiples

I: Notions Intuitives

. Système de Gestion de Base de Données (SGBD): DATA BASE MANAGEMENT SYSTEM (DBMS)

Un système qui permet de gérer une BD partagée par plusieurs utilisateurs simultanément

- ≈ 1955 1ers systèmes : chaînes de traitement → structures de données et fichiers.
- ≈ 1962 SGBD hiérarchique et réseau : privilégient les données et accès inter-fichiers prédéfinis
- ≈ 1965 Systèmes de gestion de fichiers : méthodes et langages d'analyse.
- ≈ 1970 le modèle de données relationnel de Codd; le modèle Entité/Association de Chen Sémantique de données et Normalisation des SD
- ≈ 1980 Commercialisation des SGBD relationnels Facilité d'emploi et Indépendance données / programmes
- ≈ 1985 Commercialisation des SGBD Orientés Objet

I: Notions Intuitives

. Système de Gestion de Base de Données (SGBD): DATA BASE MANAGEMENT SYSTEM (DBMS)

SGBD axés sur les données et leur qualité

- Mise à jour
- Protection
- Cohérence
- Structuration
- Irredondance
- Sémantique des données

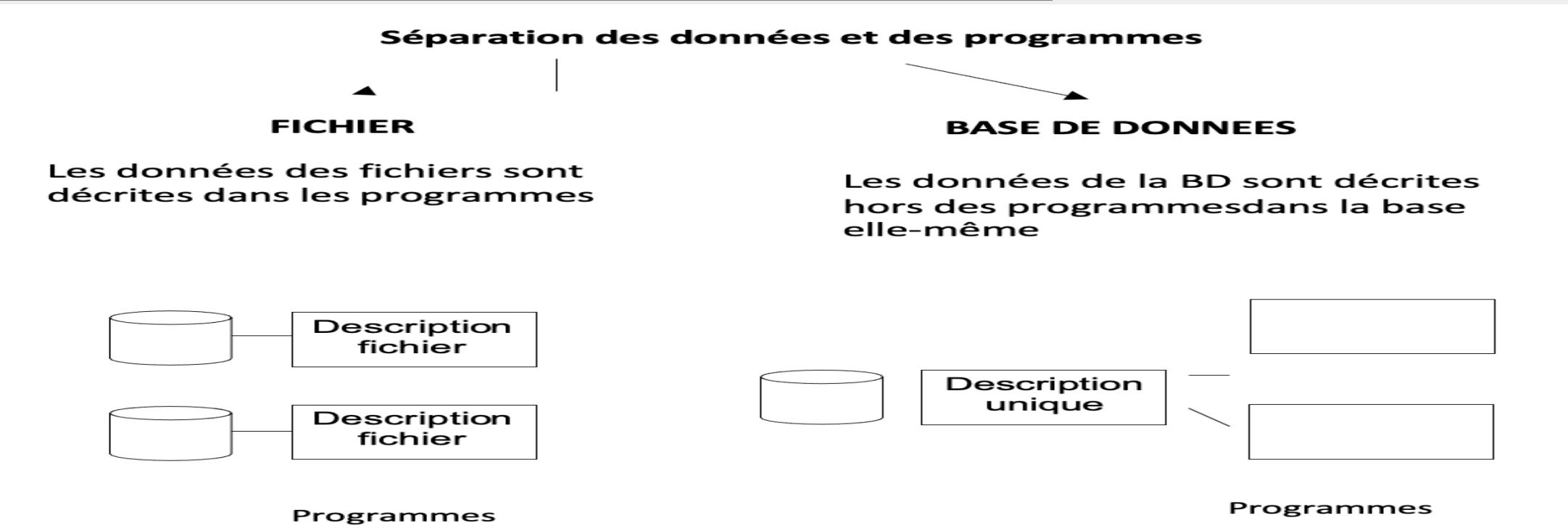
Les BD sont sous la surveillance d'un Administrateur des données

Indépendance physique avec les supports ⇒ Couche intermédiaire

Supports théoriques importants

I: Notions Intuitives

• Des fichiers aux Base de Données



La multiplication des fichiers entraînait la redondance des données, ce qui rendait difficile les mises à jour.

D'où l'idée **d'intégration** et de **partage** des données

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Décrire les données**

- Langage de définition des données (L.D.D.)
- Objets / Entités de la base (Produits, Clients ...)
Liens entre entités (1 client commande 1 produit)
- Attributs / Propriétés (Produit : réf, nom, prix ...)
- Contraintes (prix > 0)

Indépendamment des applications (de manière intrinsèque)

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Manipuler les données**

⇒ Langage de manipulation des données (L.M.D.)

- Ouverture
- Lecture ⇒ Interrogation
- Ecriture ⇒ Mise à jour
- Suppression ⇒ Mise à jour
- Fermeture

★ Distinction entre :

Langage d'interrogation (recherche)

Langage de manipulation (modification)

Interroger et mettre à jour les données sans préciser d'algorithme d'accès
dire QUOI sans dire COMMENT langage de *requêtes déclaratif*
ex: quels sont les noms des produits de prix < 100F ?

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Utilisation**

- Programme ⇒ Interface : langage hôte ↔BD
- Transactionnelle : ex. : "presse-bouton"= Unité de travail composée de plusieurs actions élémentaires, Atomique et Indivisible
- Conversationnelle ⇒ Langage de requête

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Contrôler les données**

intégrité

vérification de contraintes d'intégrité

ex.: le salaire doit être compris entre 400F et 20000F

Qualité

Définition de contraintes; Contraintes gérées par le S.G.B.D

confidentialité

contrôle des droits d'accès, autorisation

langage de contrôle des données: DATA CONTROL LANGUAGE (DCL)

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Partage**

- une BD est partagée entre plusieurs utilisateurs en même temps
- contrôle des accès concurrents
- notion de **transaction**
- L'exécution d'une transaction doit préserver la cohérence de la BD

- **Sécurité**

- reprise après panne, journalisation

- **Performances d'accès**

- index (hashage, arbres balancés ...)

II: Objectifs et avantages des SGBD

Fonctionnalités : Que doit permettre un SGBD ?

- **Indépendance physique**

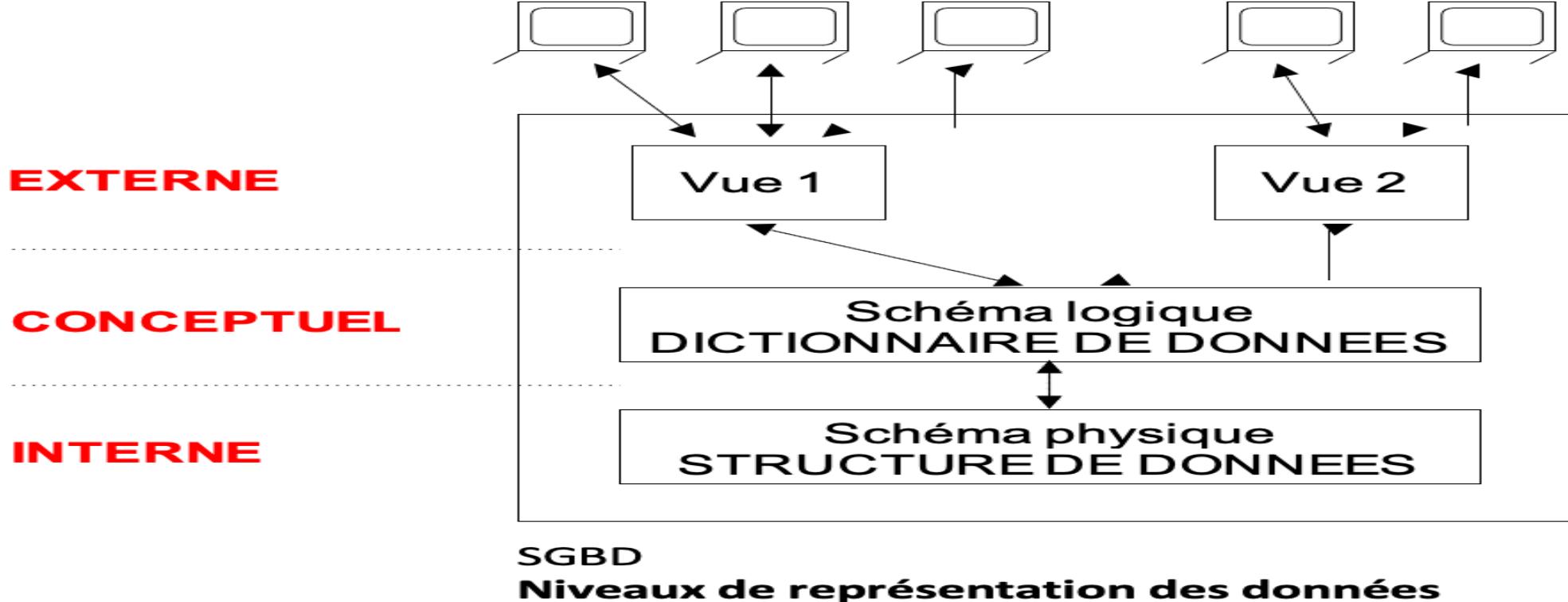
- . Pouvoir modifier les structures de stockage ou les index sans que cela ait de répercussion au niveau des applications
- Les disques, les méthodes d'accès, les modes de placement, le codage des données ne sont pas apparents

- **Indépendance logique**

- . Permettre aux différentes applications d'avoir des vues différentes des mêmes données
- Permettre au DBA de modifier le schéma logique sans que cela ait de répercussion au niveau des applications

III: Architecture ANSI/SPARC

- Proposition en 75 de l' ANSI/SPARC (*Standard Planning And Requirement Comitte*)
- 3 niveaux de représentation des données



III: Architecture ANSI/SPARC

- **Le niveau externe**
 - . Le concept de **vue** permet d'obtenir l'indépendance logique
 - La modification du schéma logique n'entraîne pas la modification des applications (une modification des vues est cependant nécessaire)
 - Chaque vue correspond à la perception d'une partie des données, mais aussi des données qui peuvent être synthétisées à partir des informations représentées dans la BD (par ex. statistiques)
- **Le niveau conceptuel**
 - . Il contient la description des données et des contraintes d'intégrité (Dictionnaire de Données)
 - Le schéma logique découle d'une activité démodélisation
- **Le niveau interne**
 - Il correspond aux structures de stockage et aux moyens d'accès (index)

Pour résumer :

Les fonctions des SGBD

⌚ DEFINITION DES DONNEES

- ***Langage de définition des données (DDL)***: (conforme à un modèle de données)

⌚ MANIPULATION DES DONNEES

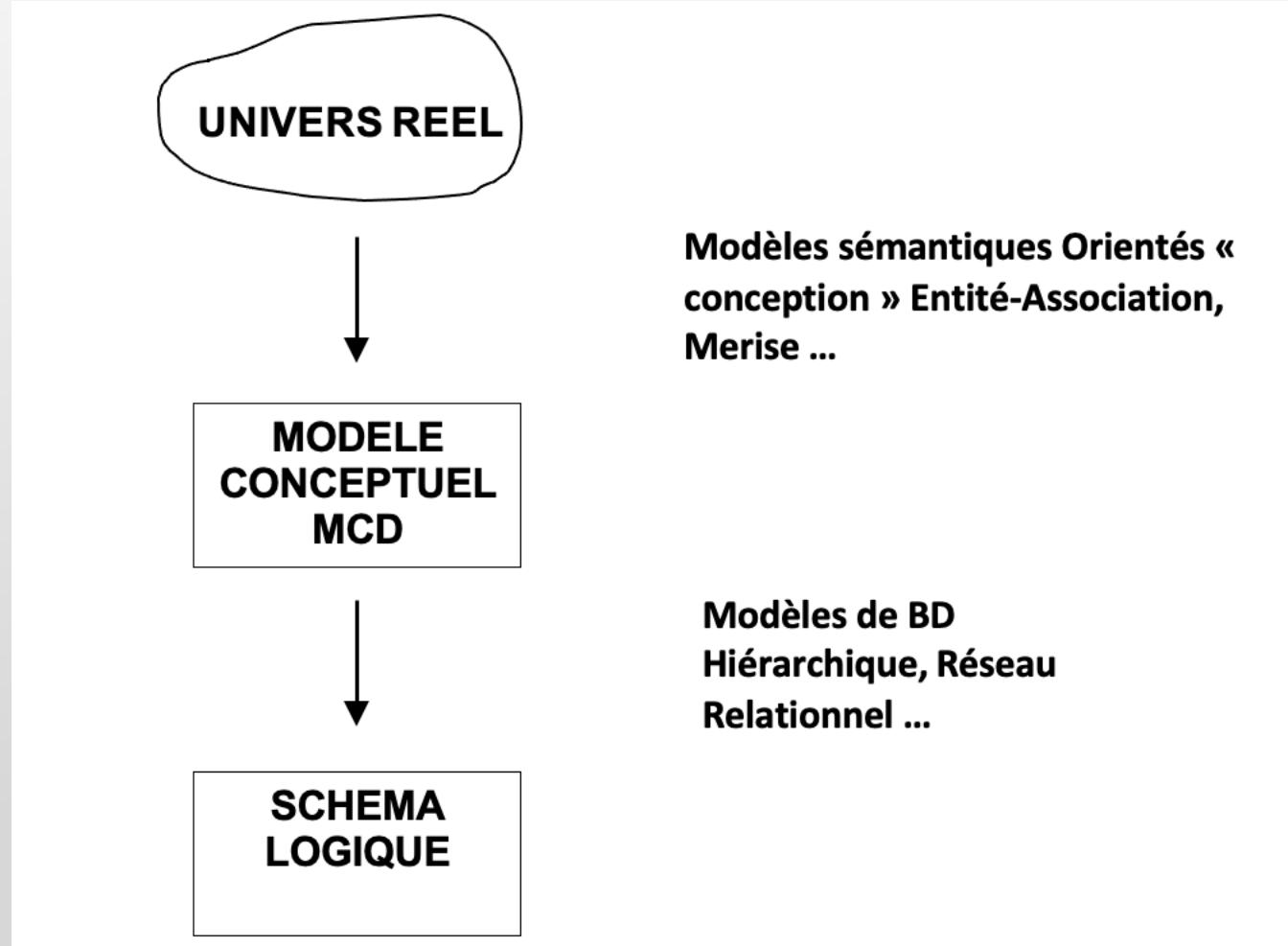
- **Interrogation**
- **Mise a jour**: insertion, suppression, modification
- ***Langage de manipulation des données (DML)***: (langage de requête déclaratif)

⌚ CONTRÔLE DES DONNEES

- Contraintes d'intégrité
- Contrôle des droits d'accès
- Gestion de transactions
- Langage de contrôle des données (DCL)

IV: Notion de modélisation de données

- ⌚ Les modèles de BD sont souvent trop limités pour pouvoir représenter directement le monde réel
- ⌚ Méthodologies de conception présentées en ACSI, SGBD2



IV: Notion de modélisation de données

- Etape Conceptuelle : Bonne représentation des phénomènes du monde réel à modéliser ⇒ Schéma Conceptuel
- Etape Logique : Prise en compte des conditions d'utilisation des données
- Etape Physique :Adaptation au S.G.B.D. utilisé ⇒ Schéma Logique ⇒ Schéma Interne

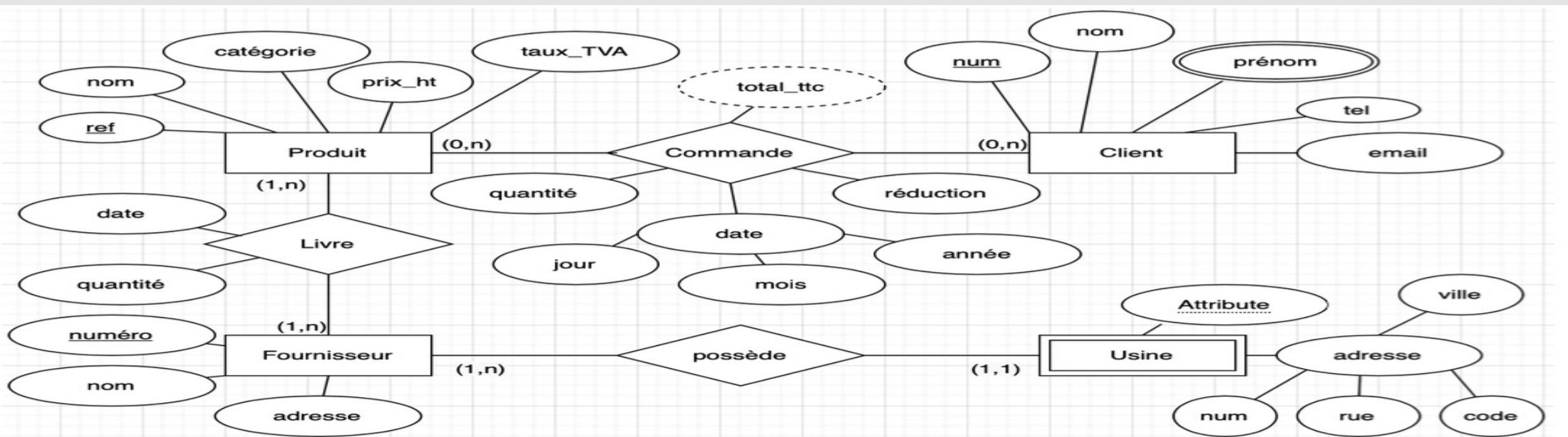
IV: Notion de modélisation de données

Le modèle Entité-Association

- EA en français, ER en anglais (pour Entity Relationship)
- Formalisme retenu par l'ISO pour décrire l'aspect conceptuel des données à l'aide d'*entités* et d'*associations*

Les concepts

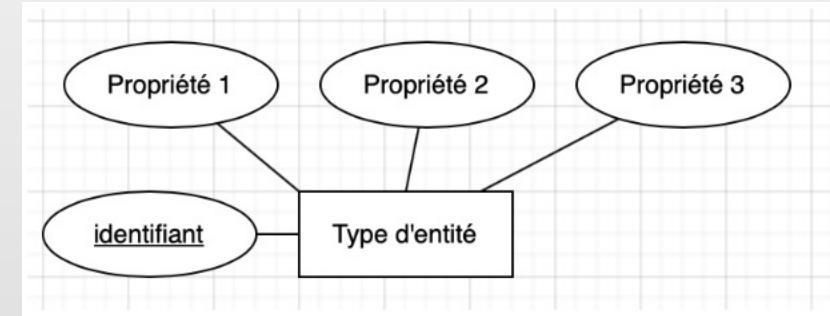
- Entité et Type d'Entité
- Association et Type de Association
- Attribut



IV: Notion de modélisation de données

Le concept d'entité

- L'entité est un objet concret ou abstrait pouvant être identifié.
- Le type d'entité est l'ensemble d'entités caractérisées par les mêmes propriétés :
 - Identifiables
 - "Distingables" dans l'ensemble
- Formellement : Ens (E, tuple) → Produit cartésien des attributs.
- Représentation graphique :



Par exemple un employé, un projet, un bulletin de paie

Nom de l'entité
Liste des propriétés

⌚ Les entités peuvent être regroupées en **types d'entités**

Par exemple, on peut considérer que tous les employés particuliers sont des **instances** du type d'entité générique EMPLOYE

Par exemple l'employé nommé DUPONT est une instance ou occurrence de l'entité EMPLOYE

IV: Notion de modélisation de données

Les propriétés

Données élémentaires relatives à une entité

Par exemple, un numéro d'employé, une date de début de projet

- ⌚ on ne considère que les propriétés qui intéressent un contexte particulier
- ⌚ Les propriétés d'une entité sont également appelées des attributs, ou des caractéristiques de cette entité

▫ L'identifiant

Propriété ou groupe de propriétés qui sert à identifier une entité

L'identifiant d'une entité est choisi par l'analyste de façon à ce que deux occurrences de cette entité ne puissent pas avoir le même identifiant

Par exemple, le numéro d'employé sera l'identifiant de l'entité EMPLOYEE

IV: Notion de modélisation de données

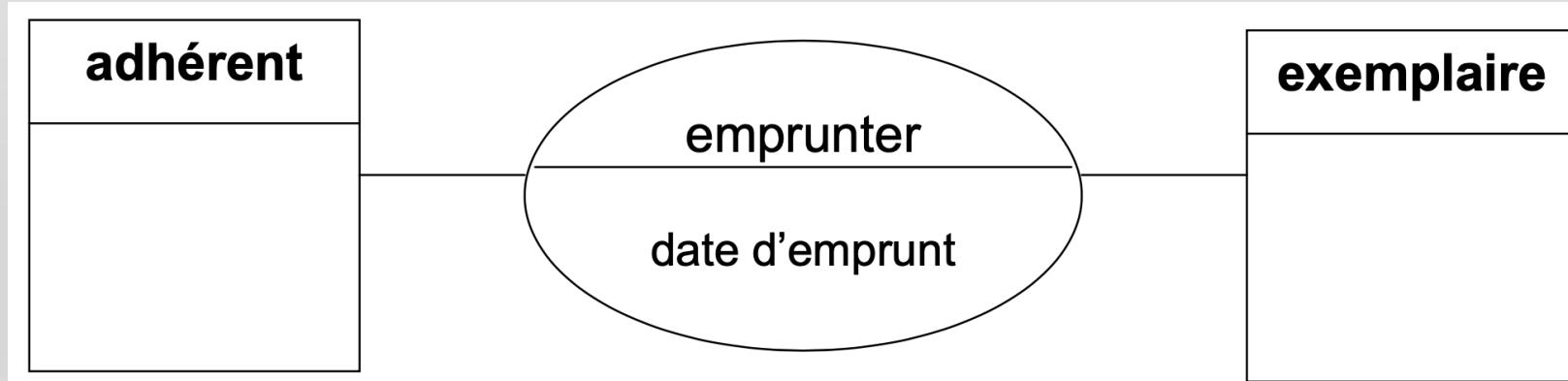
□ Les associations

Association : Ensemble d'entités dans lequel chaque entité joue un rôle bien déterminé \Leftrightarrow liaison sémantique entre entités.

Type d'association : ensemble d'associations de même sémantique définies sur les mêmes TE. Un TA a des attributs propres et des cardinalités propres.

Représentation d'un lien entre deux entités ou plus

une association peut avoir des propriétés particulières Par exemple, la date d'emprunt d'un livre

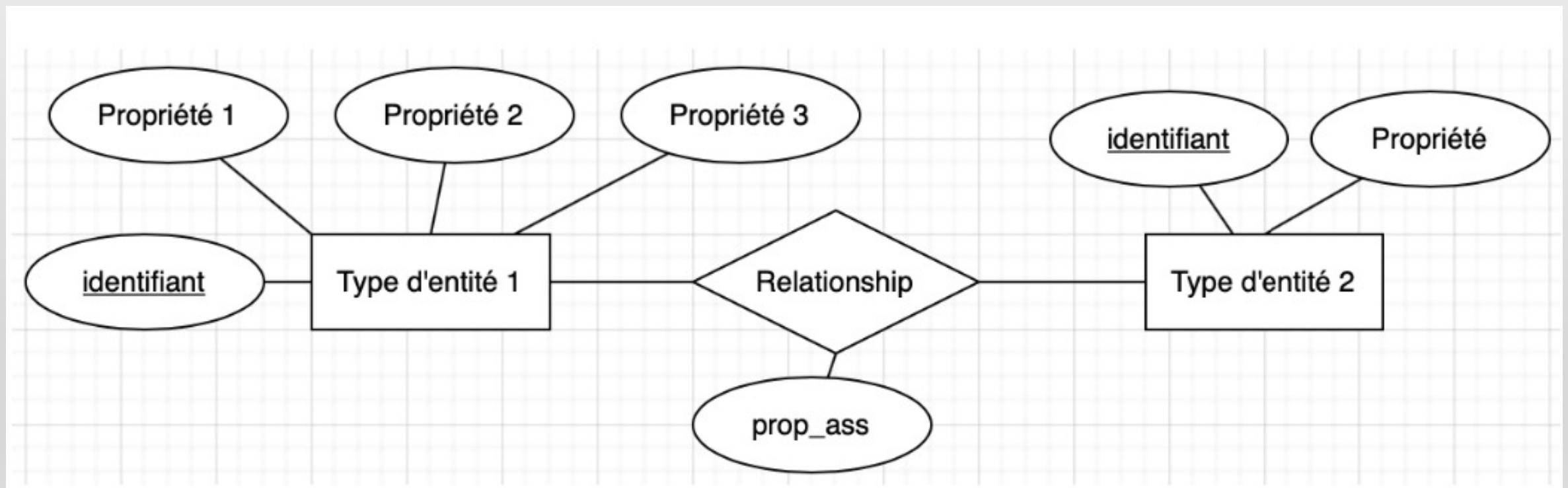


IV: Notion de modélisation de données

□ Les associations

Formellement = sous ensemble de produit cartésien : $TE1 \times TE2 \times prop_ass$

Représentation graphique :



IV: Notion de modélisation de données

□ L'attribut

Description d'une propriété; d'un type d'association

- Formellement :
Attr : TE \rightarrow Dom-Attr
- Attr : TA \rightarrow Dom-Attr

Exemple :

- Réf : Produit \rightarrow Entier
- Date : Commande \rightarrow Date

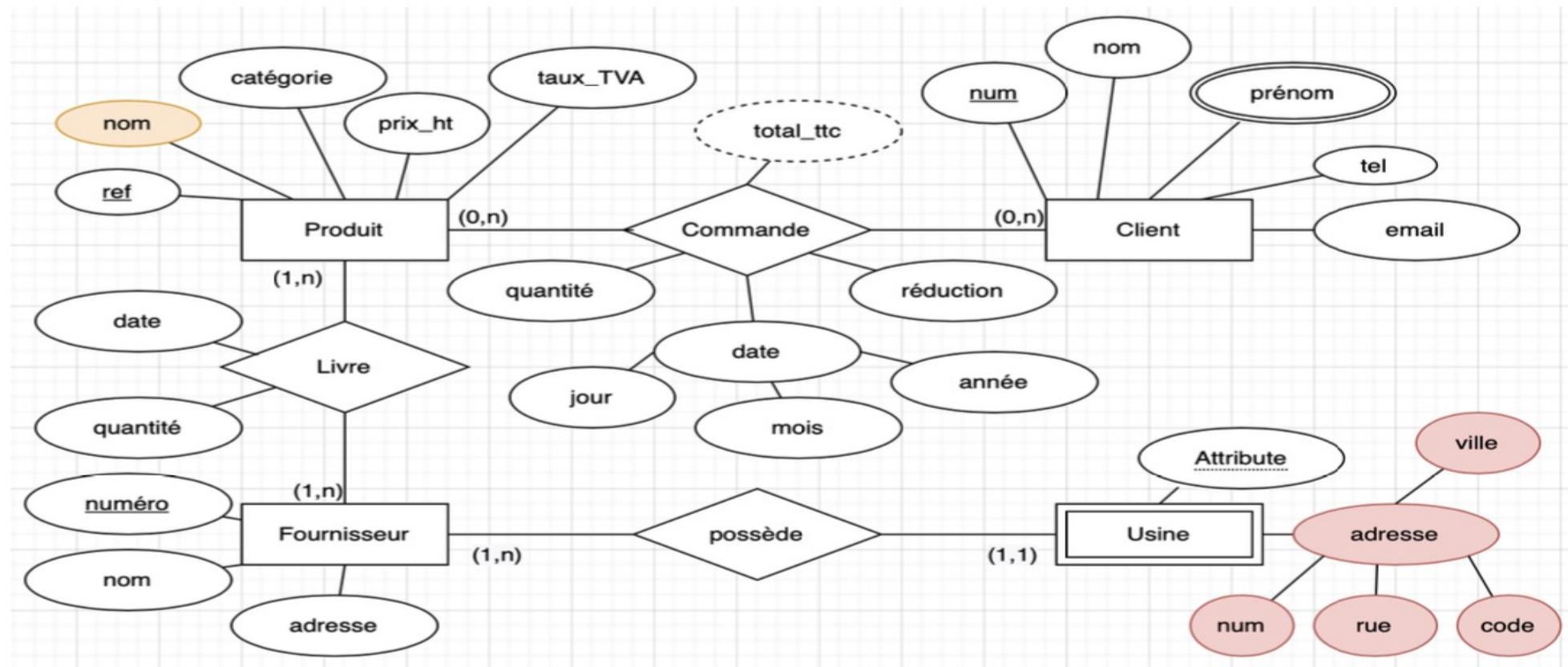
Description :

- Attr, Dom-Attr [contraintes sur le domaine]
- Exemple : Prix, Entier [prix > 0]

IV: Notion de modélisation de données

L'attribut

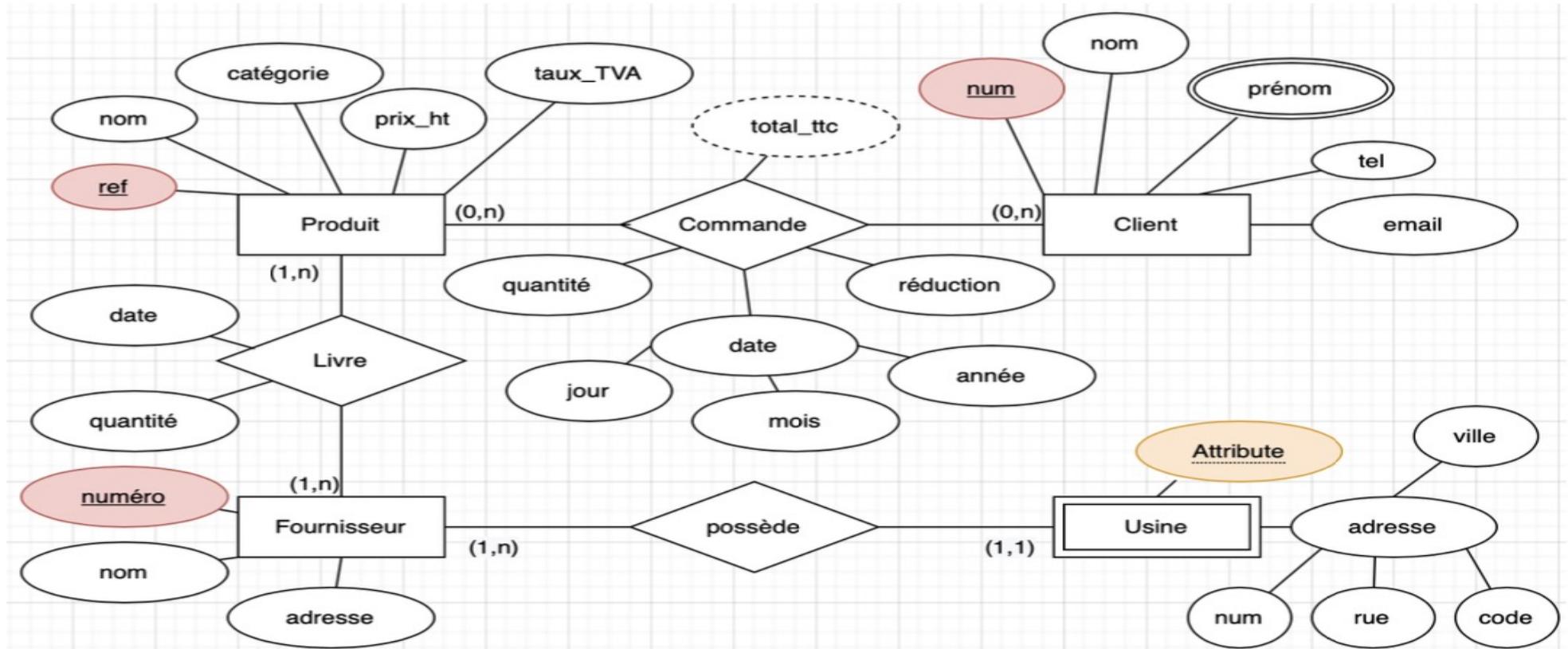
Un attribut peut être :
Simple ou composite



IV: Notion de modélisation de données

□ L'attribut

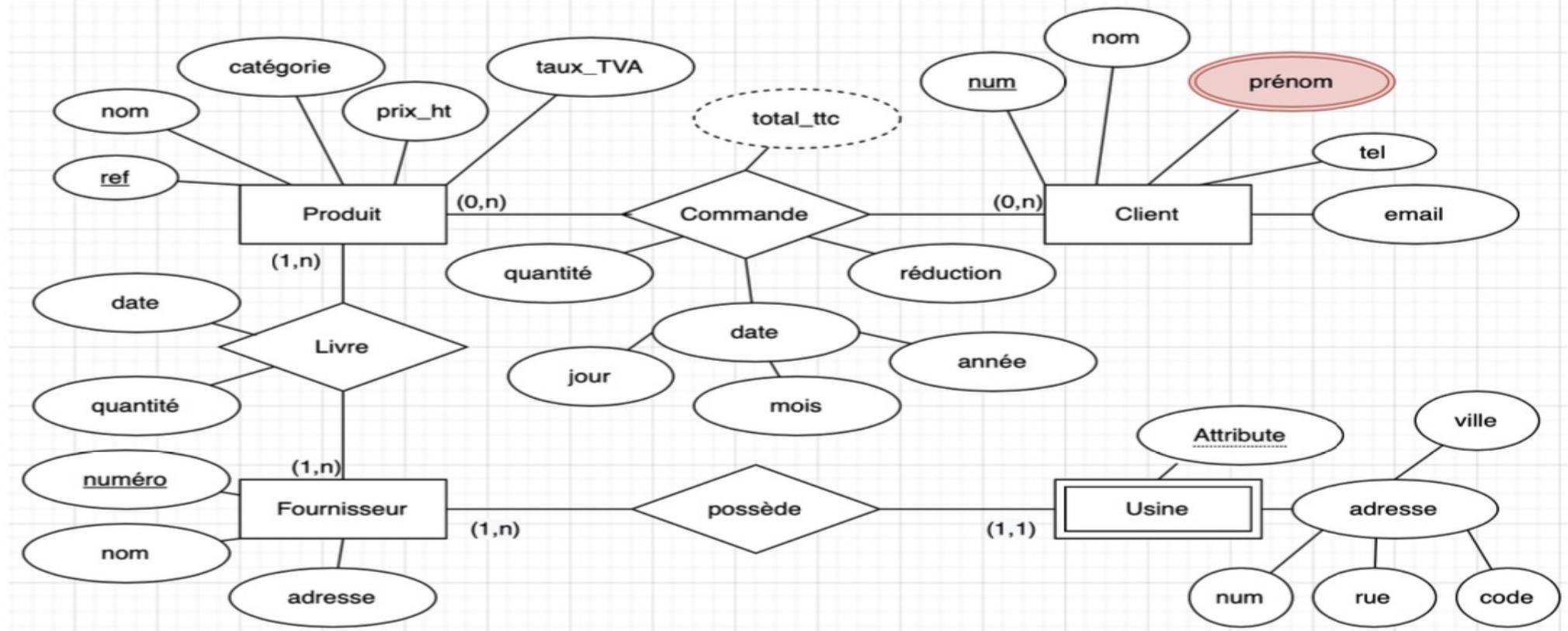
Un attribut peut être :
Identifiant



IV: Notion de modélisation de données

□ L'attribut

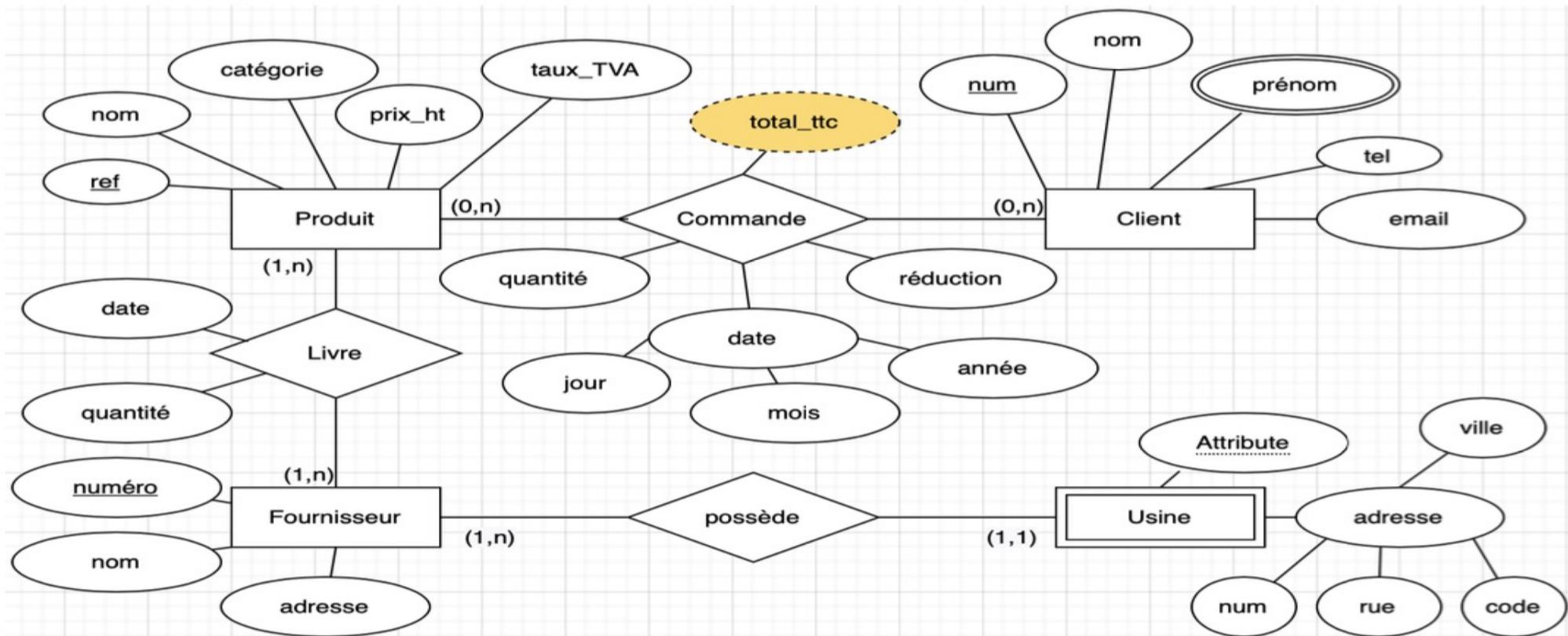
Un attribut peut être :
Simple ou multivalué



IV: Notion de modélisation de données

■ L'attribut

Un attribut peut être :
De base ou dérivé



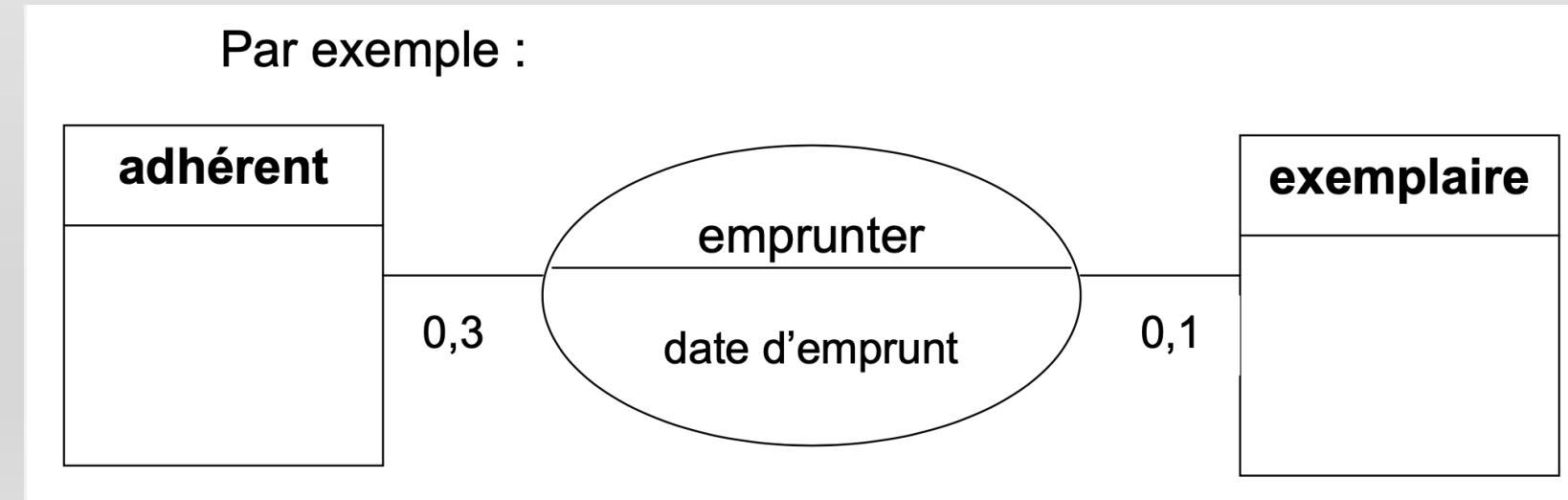
IV: Notion de modélisation de données

□ Les cardinalités

La cardinalité d'une association pour une entité constituante est constituée d'une borne minimale et d'une borne maximale :

- ⌚ Minimale : nombre minimum de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 0 ou 1
- ⌚ Maximale : nombre maximum de fois qu'une occurrence de l'entité participe aux occurrences de l'association, généralement 1 ou n

Par exemple :



IV: Notion de modélisation de données

□ Les cardinalités

- La cardinalité 0,3 indique qu'un adhérent peut être associé à 0, 1, 2 ou 3 livres, c'est à dire qu'il peut emprunter au maximum 3 livres.
- A l'inverse un livre peut être emprunté par un seul adhérent, ou peut ne pas être emprunté.

(x,y) : x représente le minimum et y le maximum de fois qu'une même occurrence/entité d'un TE peut participer à l'association.

- x = 0 ⇒ association partielle sinon elle est stable
- y = 1 ⇒ association monovaluée sinon elle est multivaluée.

3 types d'associations

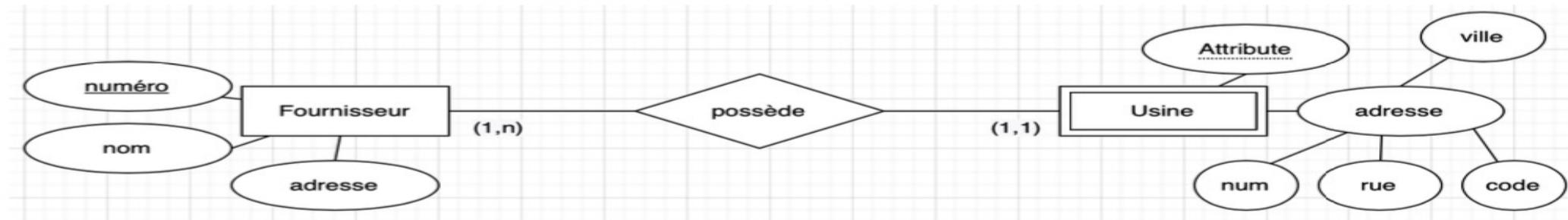
- One-to-One : (1, 1)
- One-to-Many : (1, n)
- Many-to-Many : (n, m)

IV: Notion de modélisation de données

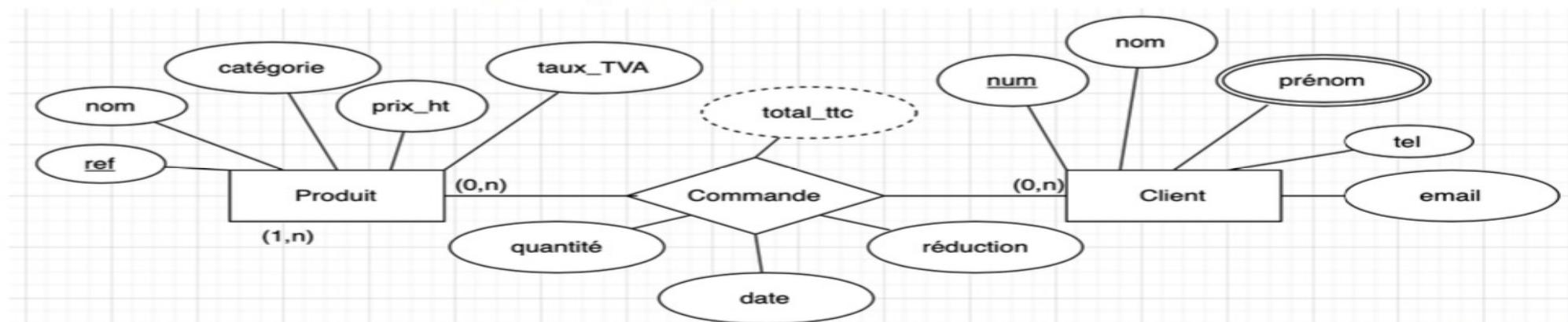
Les cardinalités

Exemple

- 3 types d'associations
 - One-to-One : (1, 1)



- One-to-Many : (1, n)

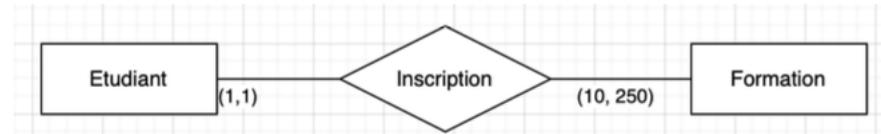


IV: Notion de modélisation de données

□ Les cardinalités

Exemple

- 3 types d'associations
 - Many-to-Many : (n, m)



- ⌚ Les cardinalités maximum sont nécessaires pour concevoir le schéma de la base de données
- ⌚ Les cardinalités minimums sont nécessaires pour exprimer les contraintes d'intégrité

En notant uniquement les cardinalités maximum, on distingue 3 types de liens :

⌚ **Lien fonctionnel 1:n**

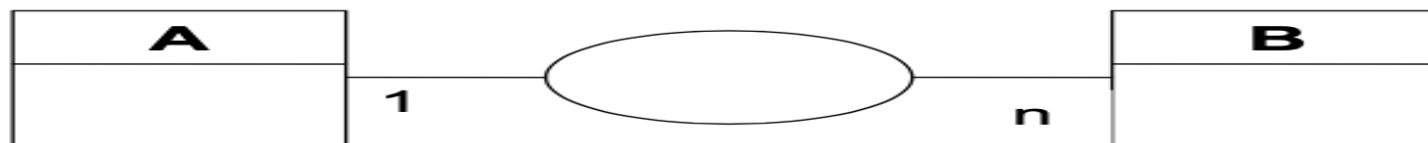
⌚ **Lien hiérarchique n:1**

⌚ **Lien maillé n:m**

IV: Notion de modélisation de données

Les cardinalités

Lien fonctionnel 1:n



Une instance de A ne peut être associée qu'à une seule instance de B

Par exemple :

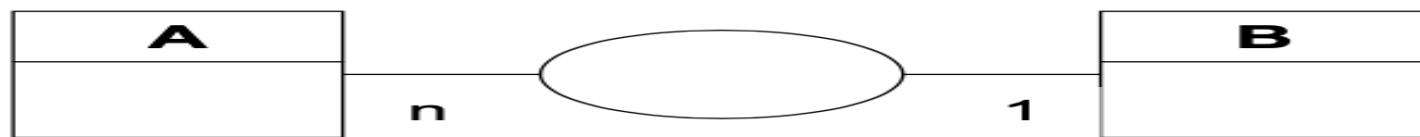


Un employé ne peut travailler que dans un seul département

IV: Notion de modélisation de données

Les cardinalités

Lien hiérarchique n:1



Une instance de A peut être associée à plusieurs instances de B

Inverse d'un lien 1:n

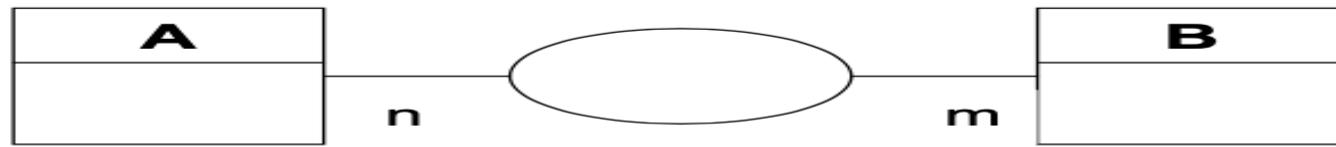


Un département emploie généralement plusieurs employés

IV: Notion de modélisation de données

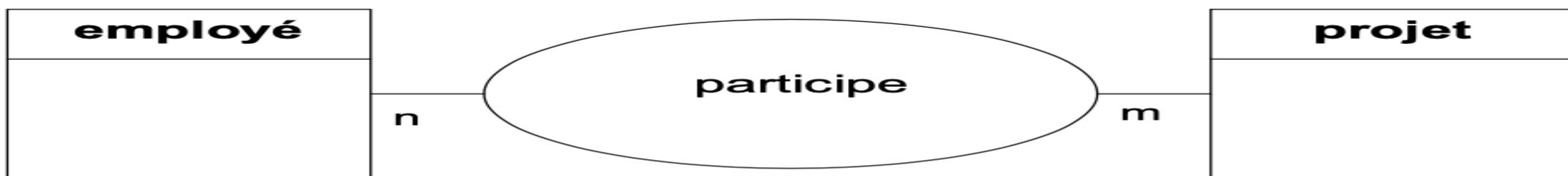
Les cardinalités

Lien maillé n:m



Une instance de A peut être associée à plusieurs instances de B et inversement

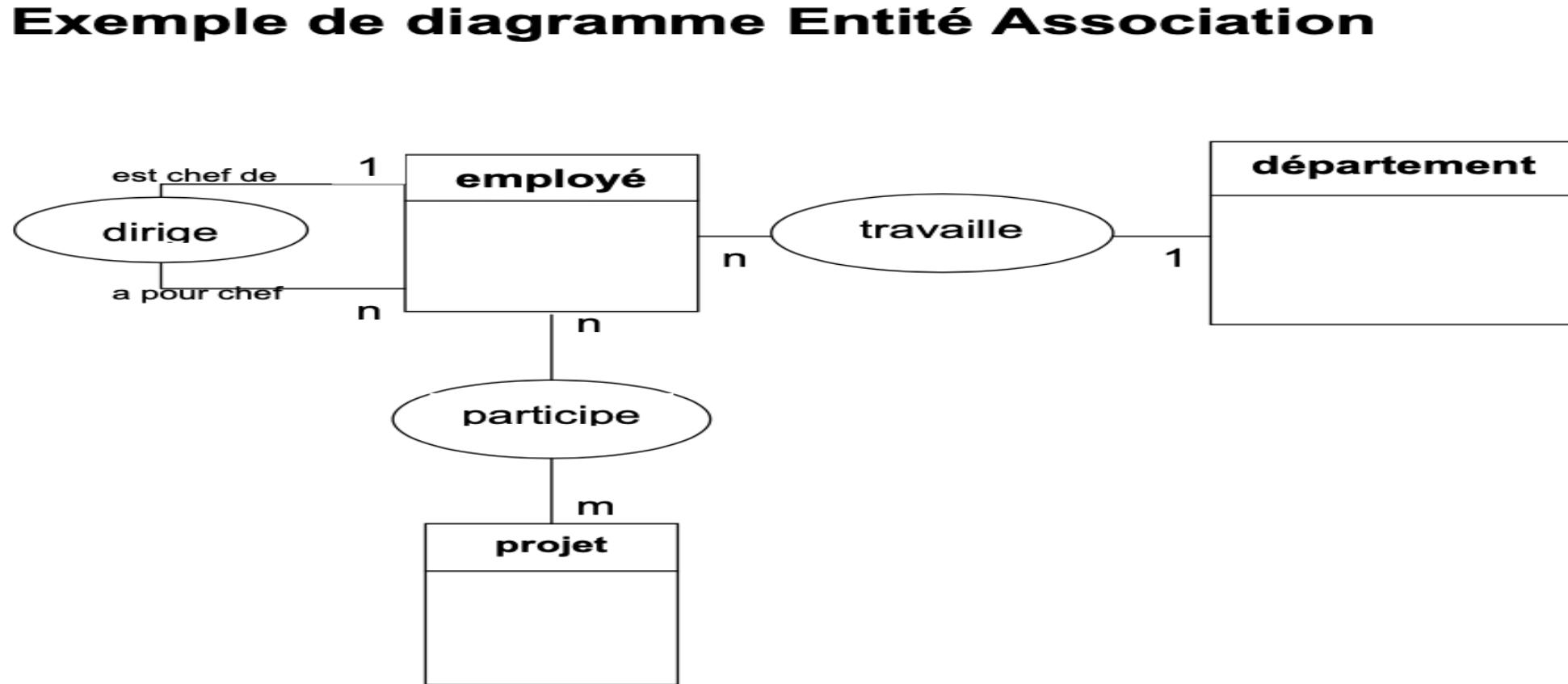
Par exemple :



De ce schéma, on déduit qu'un employé peut participer à plusieurs projets.

IV: Notion de modélisation de données

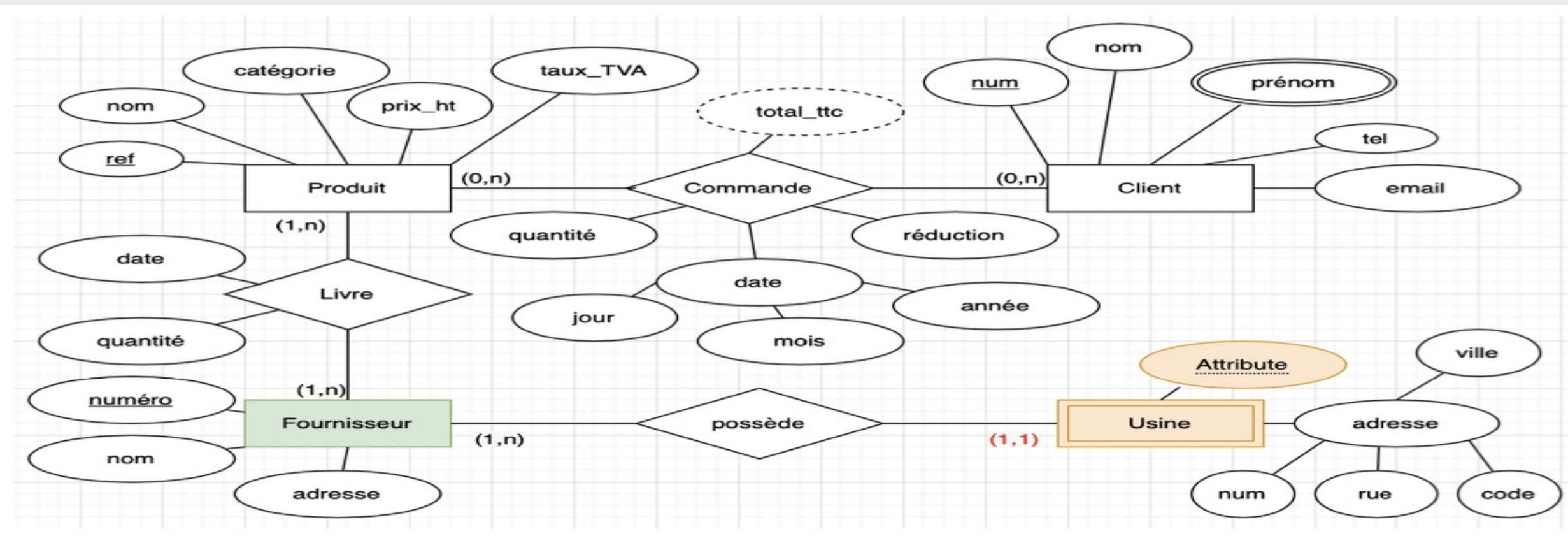
Exemple de diagramme Entité Association



IV: Notion de modélisation de données

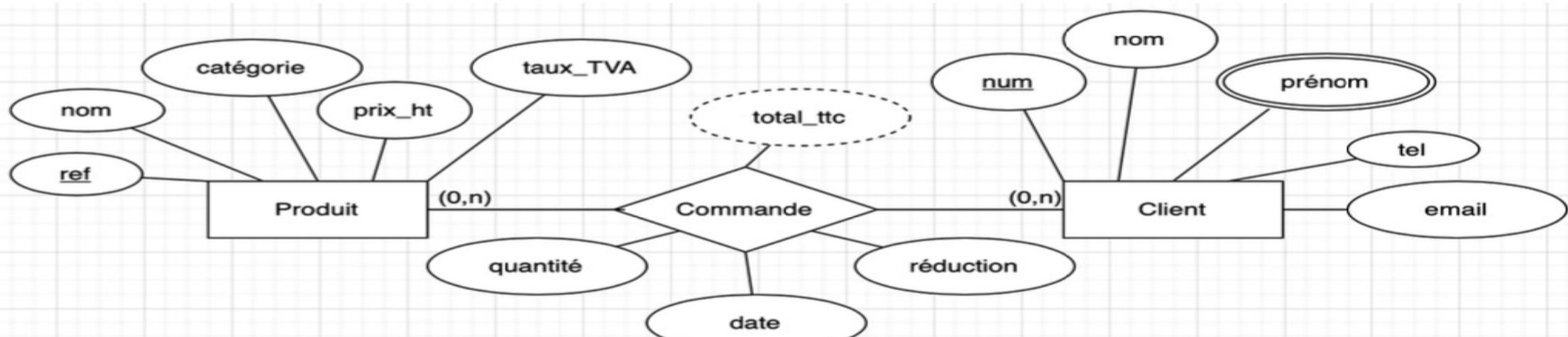
L'entité faible

L'identifiant d'une entité faible ne permet pas de l'identifier, on a également besoin de l'identifiant de l'entité forte avec laquelle elle est reliée.



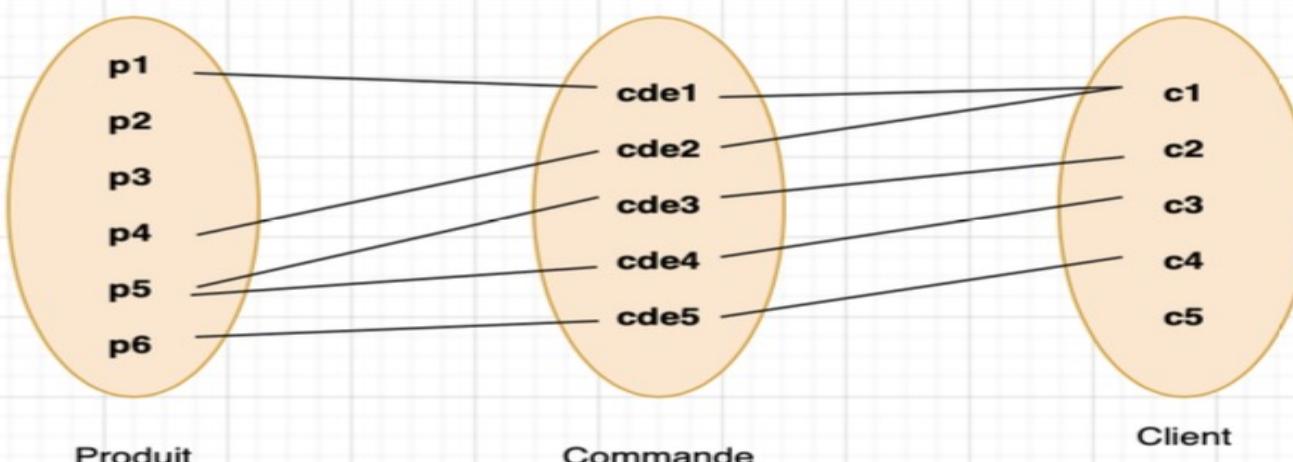
IV: Notion de modélisation de données

L'entité associative



L'identifiant de l'association Commande est (ref, num)

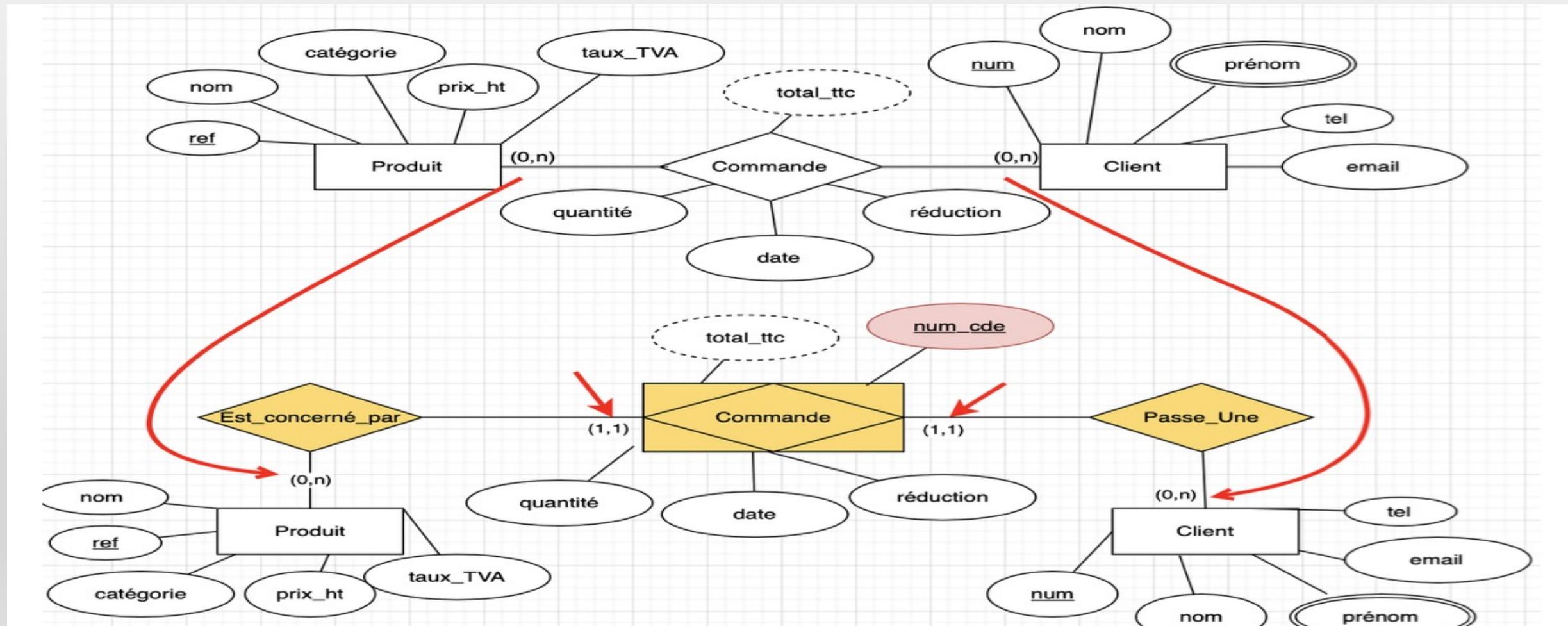
Un client ne peut pas commander deux fois le même produit



IV: Notion de modélisation de données

L'entité associative

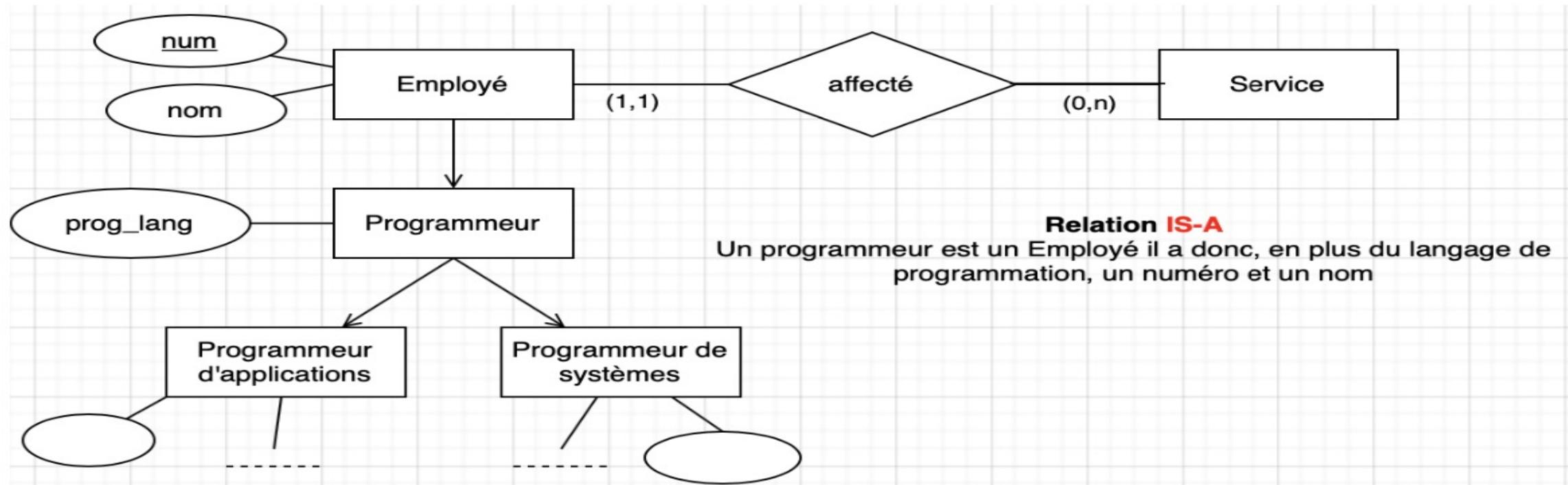
Solution : L'entité Associative qui ne peut exister que si deux entités sont associées



IV: Notion de modélisation de données

Les entités Soustype et Supertype

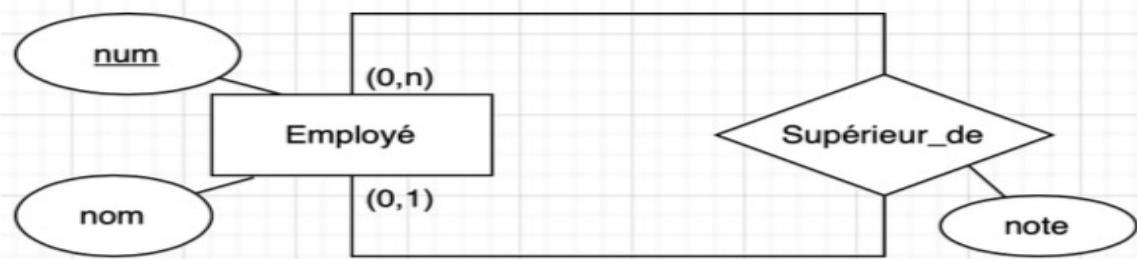
Notion aussi appelée Spécialisation/Généralisation



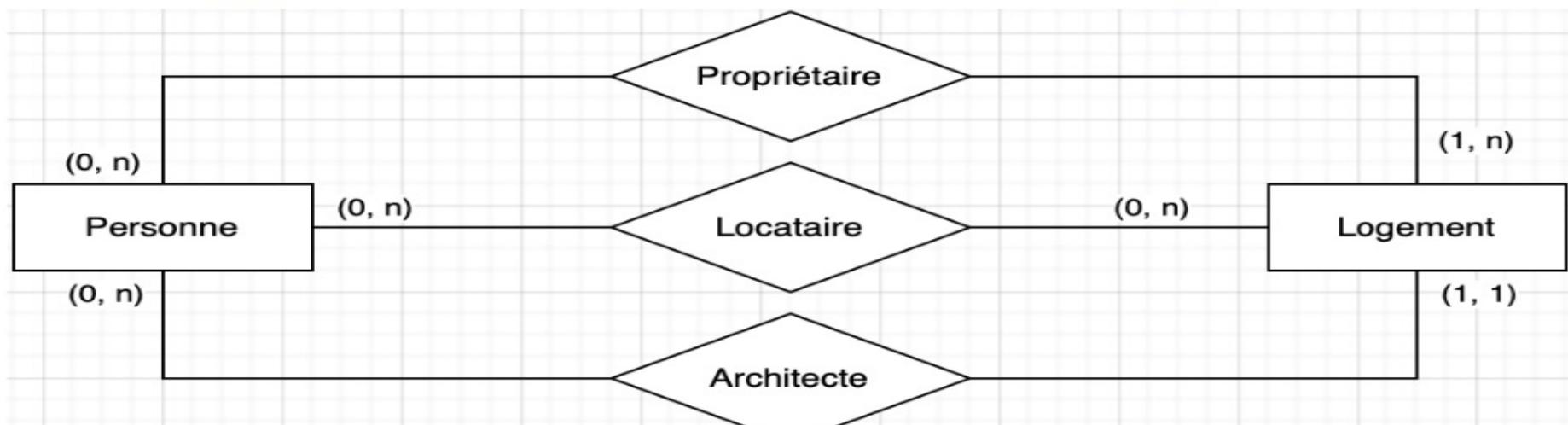
IV: Notion de modélisation de données

Types d'associations particuliers

- Type d'association définie sur un seul type d'entité : Reflexive



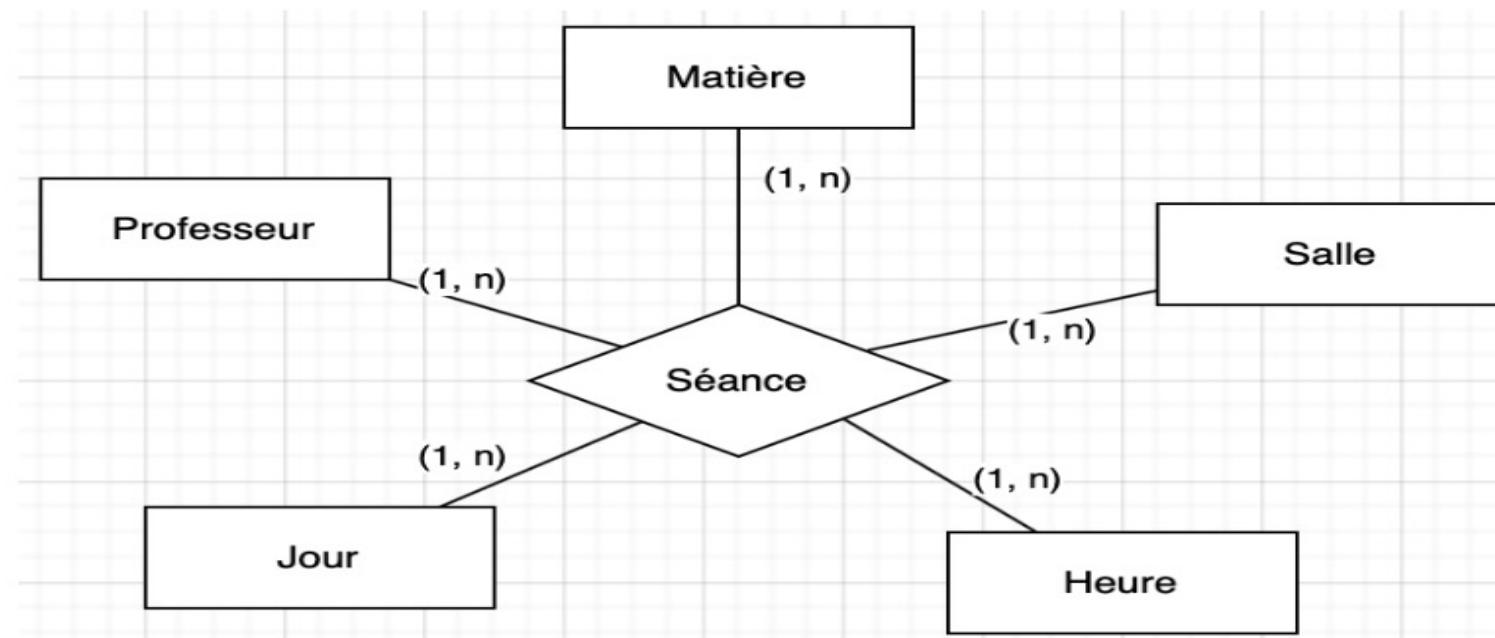
- Plusieurs types d'association entre 2 mêmes types d'entité :



IV: Notion de modélisation de données

Types d'associations particuliers

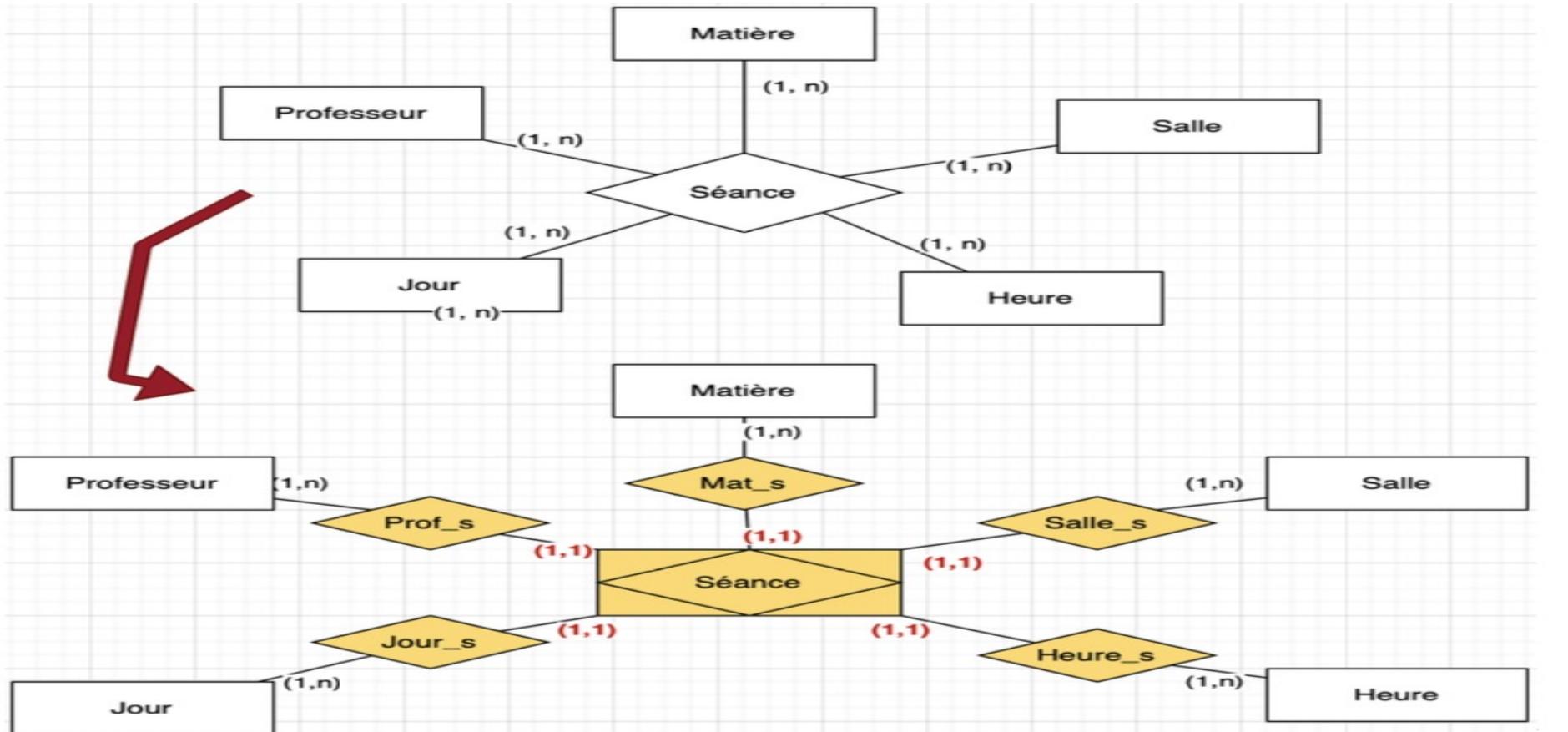
A éviter !!



IV: Notion de modélisation de données

Types d'associations particuliers

La solution



IV: Notion de modélisation de données

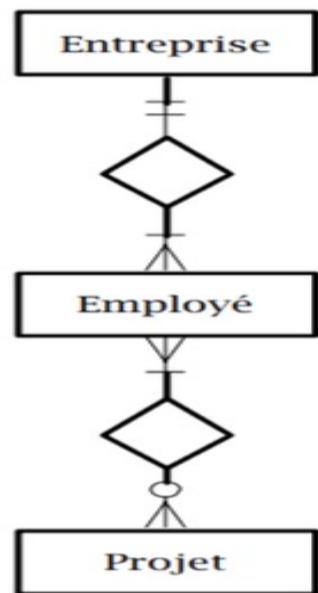
Remarque sur la notation des cardinalités

Information Engineering	Chen-1	Abrial
	— 1 —	— (1,1) —
		— (0,1) —
		— (1,n) —
	— 1 — N —	— (0,n) —
		— (0,1) —
		— (1,n) —
		— (0,n) —
		— (1,n) —
		— (0,n) —
	— M — N —	— (0,n) —

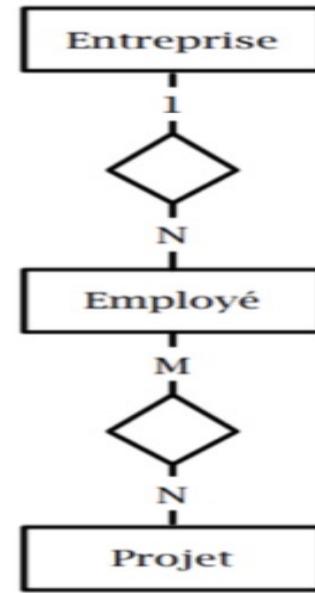
IV: Notion de modélisation de données

Remarque sur la notation des cardinalités

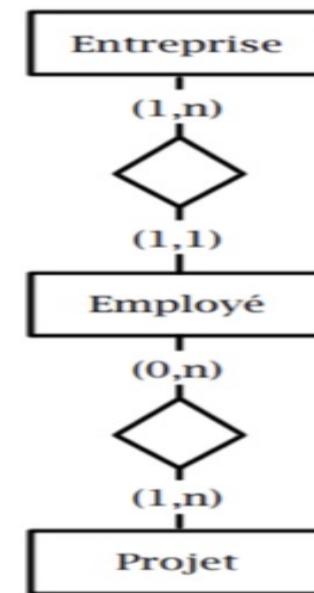
Information
Enginiering



Chen-1



Abrial



IV: Notion de modélisation de données

LE MODÈLE RELATIONNEL

En 1970, CODD présente le modèle relationnel

- Schéma logique représenté par des RELATIONS

LE SCHÉMA RELATIONNEL

Le schéma relationnel est l'ensemble des RELATIONS qui modélisent le monde réel

- Les relations représentent les entités du monde réel (comme des personnes, des objets, etc.) ou les associations entre ces entités
- Passage d'un schéma conceptuel E-A à un schéma relationnel
- une entité est représentée par la relation :
- nom_de_l'entité (liste des attributs de l'entité)
- - une association M:N est représentée par la relation :
- nom_de_l'association (liste des identifiants des entités participantes, liste des attributs de l'association)

IV: Notion de modélisation de données

LE MODÈLE RELATIONNEL

- Ex . :

CLIENT (IdCli, nom, ville)

PRODUIT (IdPro, nom, prix, qstock)

VENTE (IdCli, IdPro, date, qte)

Représentation des données sous forme de **tables** :

CLIENT

	IdCli	Nom	Ville
X		Smith	Paris
Y		Jones	Paris
Z		Blake	Nice

PRODUIT

	IdPro	Nom	Prix	Qstock
P		Auto	100	10
Q		Moto	100	10
R		Velo	100	10
S		Pedalo	100	10

VENTE

	IdCli	IdPro	Date	Qte
X		P		1
X		Q		2
X		R		3
Y		P		4
Y		Q		5
Z		Q		6

IV: Notion de modélisation de données

LES AVANTAGES DU MODÈLE RELATIONNEL

- **SIMPLICITE DE PRÉSENTATION**

- représentation sous forme de tables

- **OPÉRATIONS RELATIONNELLES**

- algèbre relationnelle

- langages assertionnels

- **INDEPENDANCE PHYSIQUE**

- optimisation des accès

- stratégie d'accès déterminée par le système

- **INDEPENDANCE LOGIQUE**

- concept de VUES

- **MAINTIEN DE L'INTEGRITÉ**

- contraintes d'intégrité définies au niveau du schéma

Chapitre 2 Le Model Relationnel

- LES CONCEPTS
- LES DÉPENDANCES FONCTIONNELLES
- LES RÈGLES D'INTÉGRITÉ
- LES FORMES NORMALES

I les Concepts

LE DOMAINE

- LA RELATION
- LES N-UPLETS
- LES ATTRIBUTS
- LE SCHÉMA D'UNE RELATION
- LE SCHÉMA D'UNE BDR
- LA REPRÉSENTATION

I les Concepts

LE DOMAIN

Ensemble de valeurs atomiques d'un certain type sémantique

Ex. :

NOM_VILLE = { Nice, Paris, Rome }

- les domaines sont les ensembles de valeurs possibles
- dans lesquels sont puisées les données
- deux ensembles peuvent avoir les mêmes valeurs bien que sémantiquement distincts
- Ex. :

NUM_ELV = { 1, 2, ..., 2000 } **NUM_ANNEE** = { 1, 2, ..., 2000 }

I les Concepts

LA RELATION

Sous ensemble du produit cartésien de plusieurs domaines

$$R \subset D_1 \times D_2 \times \dots \times D_n$$

D_1, D_2, \dots, D_n sont les domaines de R n est le degré ou l'arité de R

Ex.:

Les domaines :

$$\text{NOM_ELV} = \{ \text{dupont}, \text{durant} \}$$

$$\text{PREN_ELV} = \{ \text{pierre}, \text{paul}, \text{jacques} \} \quad \text{DATE_NAISS} = \{ \text{Date entre 1/1/1990 et 31/12/2020} \}$$

$$\text{NOM_SPORT} = \{ \text{judo}, \text{tennis}, \text{foot} \}$$

La relation ELEVE

$$\text{ELEVE} \subset \text{NOM_ELV} \times \text{PREN_ELV} \times \text{DATE_NAISS}$$

$$\text{ELEVE} = \{ (\text{dupont}, \text{pierre}, 1/1/1992),$$

$$(\text{durant}, \text{jacques}, 2/2/1994) \}$$

La relation INSCRIPT

$$\text{INSCRIPT} \subset \text{NOM_ELV} \times \text{NOM_SPORT}$$

$$\text{INSCRIPT} = \{ (\text{dupont}, \text{judo}), (\text{dupont}, \text{foot}),$$

$$(\text{durant}, \text{judo}) \}$$

I les Concepts

LES N-UPLETS

Un élément d'une relation est un n-uplet de valeurs (tuple en anglais)

- un **n-uplet** représente un fait Ex.:
« Dupont pierre est un élève né le 1 janvier1992 »
« dupont est inscrit au judo »
- **DEFINITION PRÉDICATIVE D'UNE RELATION**

Une relation peut être considérée comme un **PRÉDICAT** à n variables

$$\theta(x,y,z) \text{vrai} \Leftrightarrow (x,y,z) \in R$$

Ex. :

$$\text{est_inscrit}(\text{dupont}, \text{judo}) \Leftrightarrow (\text{dupont}, \text{judo}) \in \text{INSCRIPT}$$

LES ATTRIBUTS

Chaque composante d'une relation est un attribut

- Le nom donné à un attribut est porteur de sens
- Il est en général différent du nom de domaine
- Plusieurs attributs peuvent avoir le même domaine

Ex. :

La relation TRAJET :

TRAJET \subset NOMVILLE \times NOMVILLE

Dans laquelle la première composante représente la ville de départ VD, la deuxième composante la ville d'arrivée VA d'un trajet.

LE SCHÉMA D'UNE RELATION

Le schéma d'une relation est défini par :

- le nom de la relation
- la liste de ses attributs

On note: $R(A_1, A_2, \dots, A_n)$

Ex.:

- Extension et Intension
- L'extension d'une relation correspond à l'ensemble de ses éléments (n-uplets)

→ le terme **RELATION** désigne une extension

- L'intention d'une relation correspond à sa signification

→ le terme **SCHÉMA DE RELATION** désigne l'intention d'une relation

LE SCHÉMA D'UNE BDR

Le schéma d'une base de données est défini par :

- l'ensemble des schémas des relations qui la composent

Notez la différence entre :

- le schéma de la BDR qui dit comment les données sont organisées dans la base
- l'ensemble des n-uplets de chaque relation, qui représentent les données stockées dans la base
- **Conception de Schéma Relationnel**
- Problème:
- Comment choisir un schéma approprié ?
- Méthodologies de conception
- → cours ACSI
- cours SGBD 2

I les Concepts

LA REPRÉSENTATION

1 RELATION = 1 TABLE

U1	V1	W1	X1	Y1
U2	V2	W2	X2	Y2
U3	V3	W3	X3	Y3

1 ÉLÉMENT ou n-uplet = 1 LIGNE

LIGNE →
1 élément

U1	V1	W1	X1	Y1

* une relation est un ensemble ⇒ on ne peut pas avoir 2 lignes identiques

1 ATTRIBUT = 1 COLONNE

U1				
U2				
U3				

↑
COLONNE
1 attribut ou propriété

I les Concepts

LA REPRÉSENTATION

Exemples :

- La relation ELEVE

ELEVE :	NOM	PRENOM	NAISS
élément →	dupont	Pierre	1/1/1992
	durant	Jacques	2/2/1994
	duval	Paul	3/03/81

- La relation INSCRIPT

INSCRIPT :	NOM_ELV	SPORT
élément →	Dupont	judo
	Dupont	foot
	Durant	judo

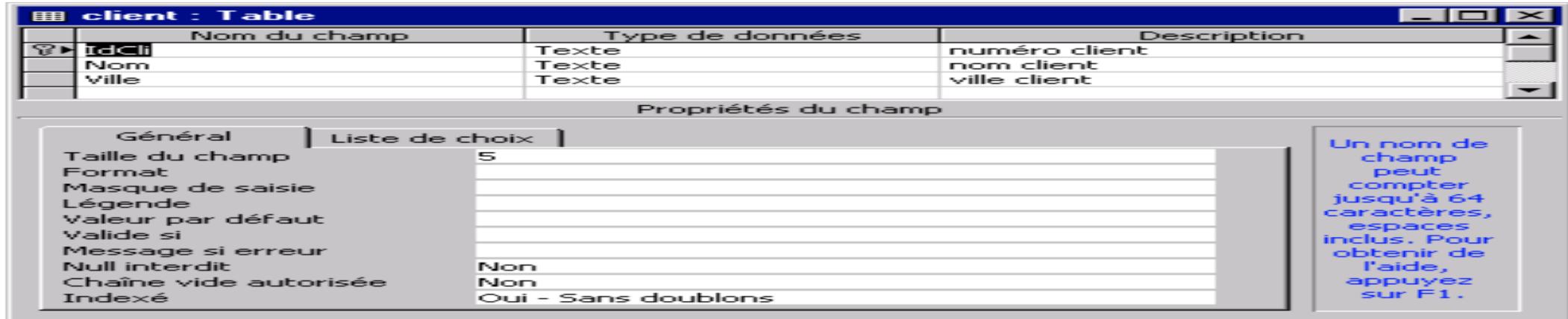
- La relation TRAJET

TRAJET :	VD	VA
élément →	Nice	paris
	Paris	rome
	Rome	nice

I les Concepts

LA REPRÉSENTATION

Fenêtre Crédation de Table d'Access



Affichage d'une table dans Access

Sélecteur d'enregistrement



II Les Dépendances fonctionnelles

Dépendance fonctionnelle

Soit R(A1, A2, ..., An) un schéma de relation

Soit X et Y des sous ensembles de {A1, A2, ..., An}

On dit que Y dépend fonctionnellement de X ($X \rightarrow Y$) si à chaque valeur de X correspond une valeur unique de Y

on écrit : $X \rightarrow Y$

on dit que : X détermine Y

Ex.:

PRODUIT (no_prod, nom, prixUHT) no_prod \rightarrow (nom, prixUHT)

NOTE (no_contrôle, no_élève, note) (no_contrôle, no_élève) \rightarrow note

- une dépendance fonctionnelle est une propriété **sémantique**, elle correspond à une contrainte supposée toujours vrai du monde réel

D.F. élémentaire

D.F. $X \rightarrow A$ mais A est un attribut unique non inclus dans X et il n'existe pas de X' inclus dans X tel que $X' \rightarrow A$

II Les Dépendances fonctionnelles

La clé d'une relation

attribut (ou groupe minimum d'attributs) qui détermine tous les autres

Ex.:

PRODUIT (**no_prod**, nom, prixUHT)
no_prod → (nom, prixUHT)
no_prod est une clé

- Une clé détermine un n-uplet de façon unique
- Pour trouver la clé d'une relation, il faut examiner attentivement les **hypothèses sur le monde réel**
- Une relation peut posséder plusieurs clés, on les appelle **clés candidates**
- Ex.:
dans la relation PRODUIT, nom est une clé candidate (à condition qu'il n'y ait jamais 2 produits de même nom)

II Les Dépendances fonctionnelles

Clé primaire

choix d'une clé parmi les clés candidates

Clé étrangère ou clé secondaire

attribut (ou groupe d'attributs) qui fait référence à la clé primaire d'une autre relation

Ex.:

CATEG (no_cat, design, tva)

PRODUIT(no_prod, nom, marque, no_cat, prixUHT)

no_cat dans PRODUIT est une clé étrangère

CLÉ ÉTRANGÈRE = CLÉ PRIMAIRE dans une autre relation

III Les Règles d'Intégrité

Les règles d'intégrité sont les assertions qui doivent être vérifiées par les données contenues dans une base

Le modèle relationnel impose les contraintes structurelles suivantes :

- **INTÉGRITÉ DE DOMAINE**
- **INTÉGRITÉ DE CLÉ**
- **INTÉGRITÉ RÉFÉRENCIELLE**
- La gestion automatique des contraintes d'intégrité est l'un des outils les plus importants d'une base de données.
- Elle justifie à elle seule l'usage d'un SGBD.

III Les Règles d'Intégrité

INTÉGRITÉ DE DOMAINE

Les valeurs d'une colonne de relation doivent appartenir au domaine correspondant

- contrôle des valeurs des attributs
- contrôle entre valeurs des attributs

INTÉGRITÉ DE CLÉ

Les valeurs de clés primaires doivent être :

- uniques
- nonNULL
- Unicité de clé
- Unicité des n-uplets
- **Valeur NULL:** valeur conventionnelle pour représenter une information **inconnue**
- dans toute extension possible d'une relation, il ne peut exister 2 n-uplets ayant même valeur pour les attributs clés sinon 2 clés identiques détermineraient 2 lignes identiques (d'après la définition d'une clé), ce qui est absurde

III Les Règles d'Intégrité

INTÉGRITÉ RÉFÉRENCIELLE

Les valeurs de clés étrangères sont 'NULL' ou sont des valeurs de la clé primaire auxquelles elles font référence

- Relations dépendantes

- LES DÉPENDANCES :

Liaisons de un à plusieurs exprimées par des attributs

particuliers: **clés étrangères** ou **clés secondaires**

III Les Règles d'Intégrité

INTÉGRITÉ RÉFÉRENCIELLE

Les contraintes de référence ont un impact important pour les opérations de mises à jour, elles permettent d'éviter les anomalies de mises à jour

Exemple :

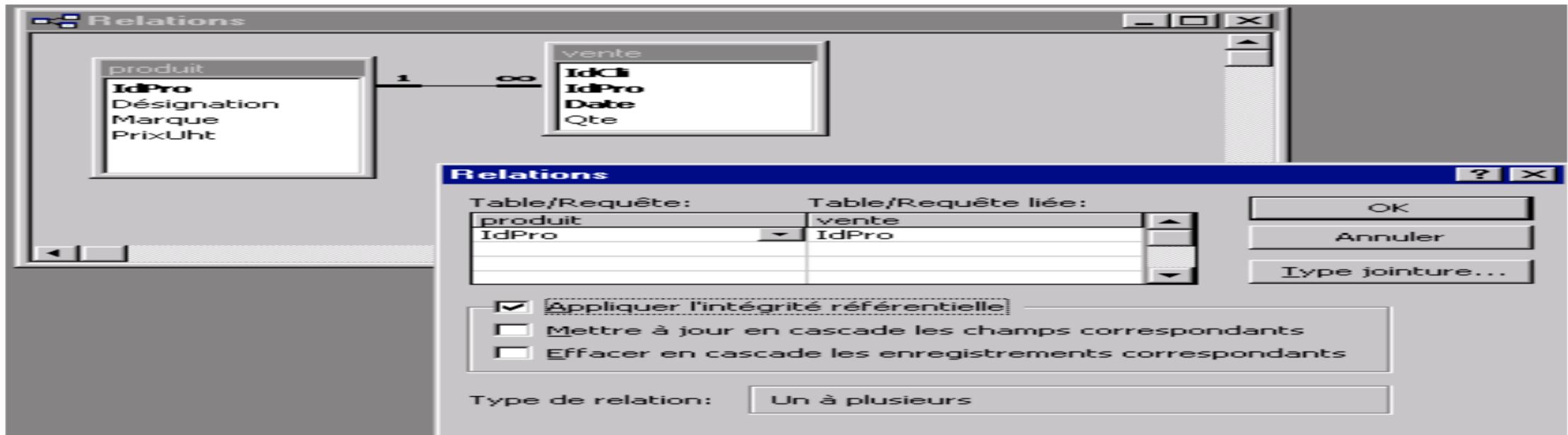
CLIENT (no_client, nom, adresse)
ACHAT (no_produit, no_client, date, qte)

Clé étrangère no_client dans ACHAT

- insertion tuple no_client = X dans ACHAT
 - ⇒ vérification si X existe dans CLIENT
- suppression tuple no_client = X dans CLIENT
 - ⇒ soit interdire si X existe dans ACHAT
 - ⇒ soit supprimer en cascade tuple X dans ACHAT
 - ⇒ soit modifier en cascade X = NULL dans ACHAT
- modification tuple no_client = X en X' dans CLIENT
 - ⇒ soit interdire si X existe dans ACHAT
 - ⇒ soit modifier en cascade X en X' dans ACHAT

III Les Règles d'Intégrité

Paramétrage des Relations dans Access



- **IdPro de Vente est une clé étrangère qui fait référence à la clé primaire de Produit**
- **Appliquer l'intégrité référentielle signifie que l'on ne pourra pas avoir, à aucun moment, une ligne de Vente avec un code produit IdPro inexistant dans la table Produit.**
- **Une valeur de clé étrangère peut être Null**

IV LES FORMES NORMALES

La théorie de la normalisation

- elle met en évidence les relations "indésirables"
- elle définit les critères des relations "désirables" appelées **formes normales**
- Propriétés indésirables des relations
 - Redondances
 - ValeursNULL
 -
- elle définit le processus de normalisation permettant de **décomposer** une relation non normalisée en un ensemble équivalent de relations normalisées

IV LES FORMES NORMALES

La décomposition

Objectif:

- décomposer les relations du schéma relationnel sans perte d'informations
- obtenir des relations canoniques ou de base du monde réel
- aboutir au schéma relationnel normalisé
- Le schéma de départ est le schéma universel de la base
- Par raffinement successifs on obtient des sous relations sans perte d'informations et qui ne seront pas affectées lors des mises à jour (**non redondance**)

Les formes normales

5 FN, les critères sont de plus en plus restrictifs

$$FN_j \Rightarrow FN_i \quad (j > i)$$

- Notion intuitive de FN

une « bonne relation » peut être considérée comme une **fonction** de la clé primaire vers les attributs restants

IV LES FORMES NORMALES

1ère Forme Normale 1FN

Une relation est en 1FN si tout attribut est atomique (non décomposable)

Contre-exemple

ELEVE (no_elv, nom, prenom, liste_notes)

Un attribut ne peut pas être un ensemble de valeurs

Décomposition

ELEVE (no_elv, nom, prenom) NOTE (no_elv, no_matiere, note)

IV LES FORMES NORMALES

2ème Forme Normale 2FN

Une relation est en 2FN si

- elle est en 1FN
- si tout attribut n'appartenant pas à la clé ne dépend pas d'une partie de la clé
- C'est la phase d'identification des clés
- Cette étape évite certaines redondances
- Tout attribut doit dépendre fonctionnellement de la totalité de la clé

•**Contre-exemple**

- une relation en 1FN qui n'est pas en 2FN
- COMMANDE** (date, no_cli, no_pro, qte, prixUHT)

elle n'est pas en 2FN car la clé = (date, no_cli, no_pro), et le **prixUHT** ne dépend que de **no_pro**

•**Décomposition**

- COMMANDE** (date, no_cli, no_pro, qte)
- PRODUIT** (no_pro, prixUHT)

IV LES FORMES NORMALES

3ème Forme Normale 3FN

Une relation est en 3FN si

- elle est en 2FN
- si tout attribut n'appartenant pas à la clé ne dépend pas d'un attribut non clé

Ceci correspond à la non transitivité des D.F. ce qui évite les redondances.

En 3FN une relation préserve les D.F. et est sans perte.

Contre-exemple

une relation en 2FN qui n'est pas en 3FN

VOITURE (matricule, marque, modèle, puissance)

on vérifie qu'elle est en 2FN ; elle n'est pas en 3FN car la clé =
matricule, et la **puissance** dépend de **(marque, modèle)**

Décomposition

VOITURE (matricule, marque, modèle)

MODELE (marque, modèle, puissance)

IV LES FORMES NORMALES

3ème Forme Normale de BOYCE-CODD BCNF

Une relation est en BCFN :

- elle est en 1FN et
- ssi les seules D.F. élémentaires sont celles dans lesquelles une clé détermine un attribut

- BCNF signifie que l'on ne peut pas avoir un attribut (ou groupe d'attributs) déterminant un autre attribut et distinct de la clé
- Ceci évite les redondances dans l'extension de la relation: mêmes valeurs pour certains attributs de n-uplets différents
- BCNF est plus fin que FN3 : **BCNF \Rightarrow FN3**
- Contre-exemple**
- une relation en 3FN qui n'est pas BCNF
- CODEPOSTAL (ville, rue, code)

on vérifie qu'elle est FN3, elle n'est pas BCNF car la clé = (ville, rue) (ou (code, ville) ou (code, rue)), et code \rightarrow ville

Chapitre 3 Présentation des données

Une fois la base et les tables créées, il faut pouvoir les exploiter.

L'utilisateur final aura besoin de visualiser et saisir des données,d'effectuer des calculs et d'imprimer des résultats.

La réponse à ces problèmes de présentation des données est fournie par :

- **les formulaires**

destinés à être affichés à l'écran

- **les états**

destinés à être imprimés.

I Les formulaires

2 types de formulaires :

- **de présentation des données**

Ils permettent de saisir, ou modifier les données d'une ou plusieurs tables sous une forme visuellement agréable

- **de distribution**

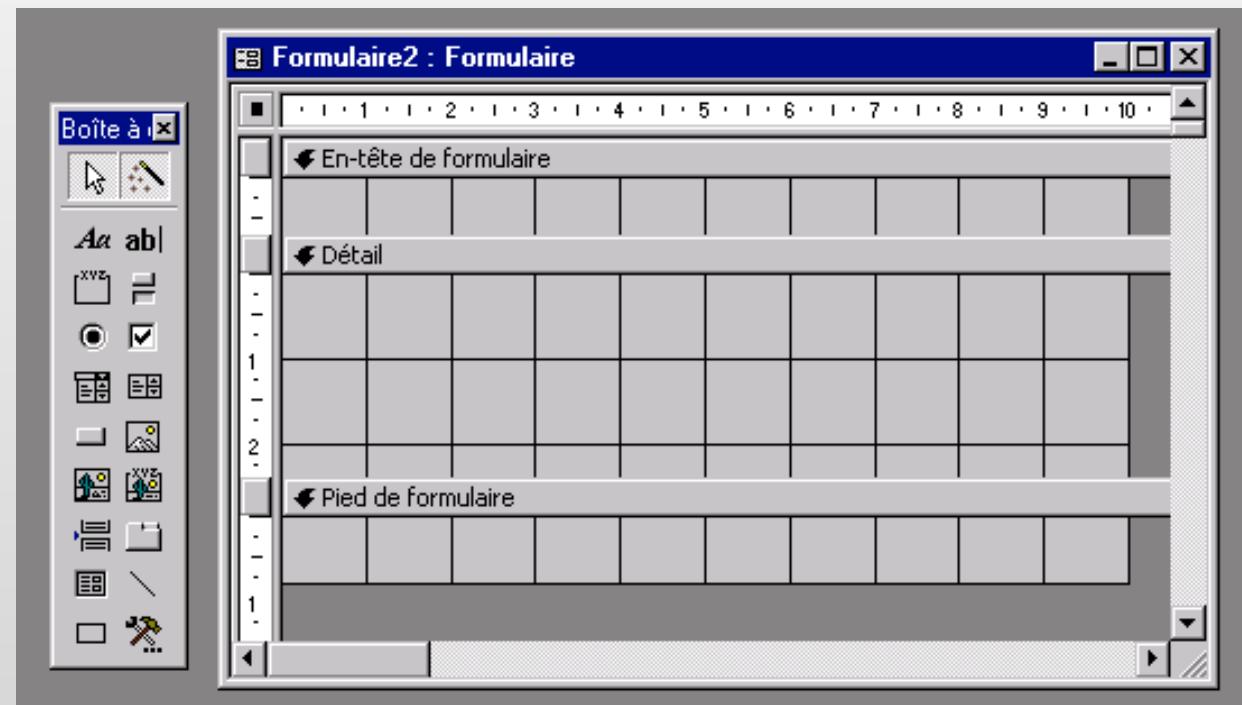
Ils ne sont attachés à aucune table, et servent uniquement de page de menu pour orienter l'utilisateur vers d'autres formulaires ou états

I | Les formulaires

Formulaire rudimentaire

The screenshot shows a Windows-style window titled "produit". Inside, there are four text input fields: "IdPro" containing "100", "Désignation" containing "ps", "Marque" containing "ibm", and "PrixUht" containing "5 000,00 F". Below these fields is a navigation bar with buttons for moving between records and a status message "Enr: 1 sur 4".

Fenêtre Conception de Formulaire d'Access



I Les formulaires

Formulaire avec sous-formulaire

IdPro	IdC	Désignation	Marque	PrixUht	Qstoc
10	C1	ps	ibm	5 000,00 F	20
20	C1	mac	apple	7 500,00 F	20
30	C1	aptiva	ibm	12 000,00 F	10
40	C1	power mac	apple	20 000,00 F	10

Permet d'afficher les données de deux tables qui sont en relation l'une avec l'autre.

- Le formulaire principal affiche les données de la table principale
- Le sous formulaire affiche les données de la table liée

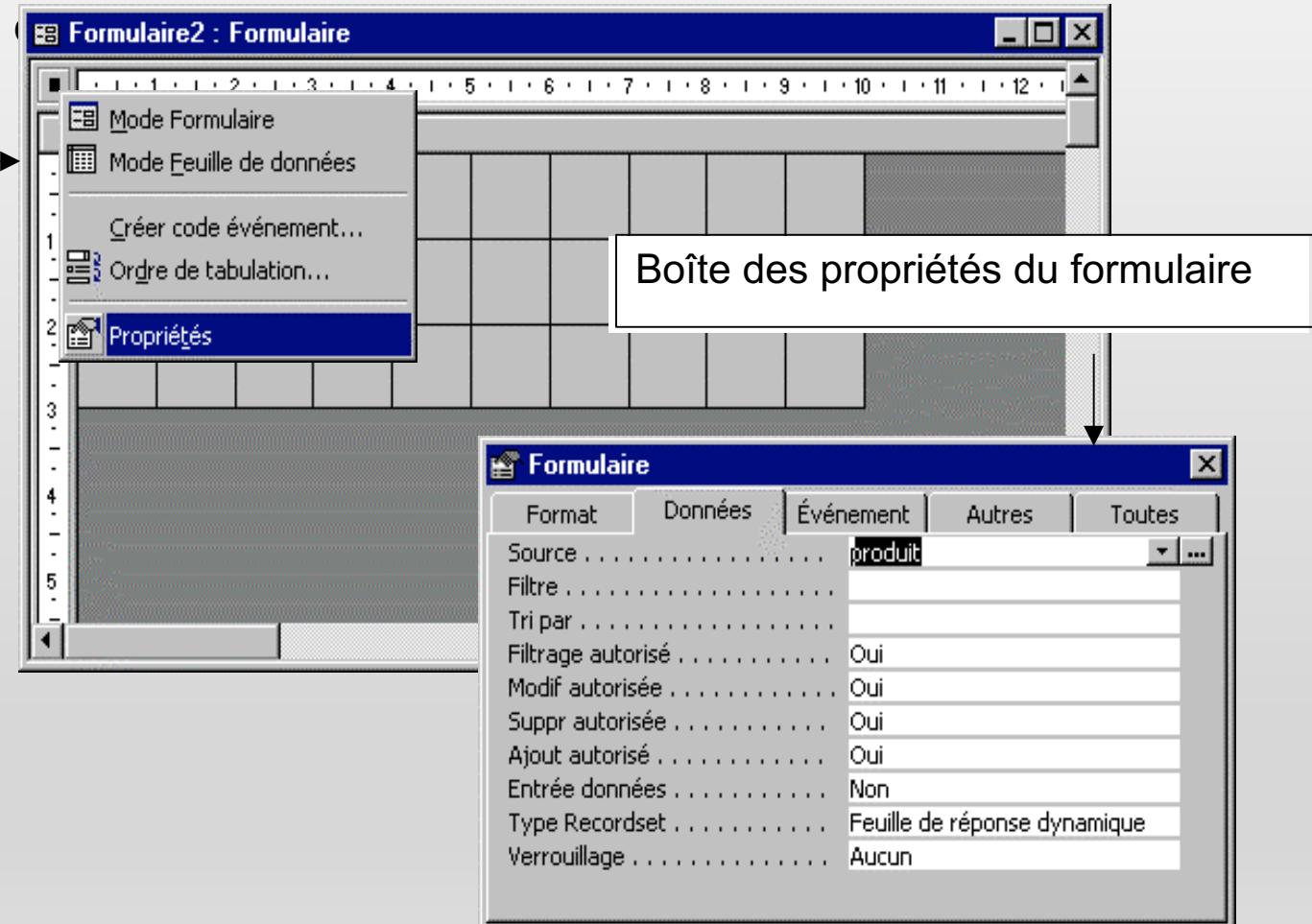
Si l'utilisateur change d'enregistrement principal, le sous formulaire est automatiquement mis à jour.

I | Les formulaires

Création d'un formulaire de présentation

- 1) Définir la propriété Source de données (table ou requête)

Cliquer ici avec le bouton droit, puis sélectionner Propriétés



- Sélectionner l'onglet Données
- Définir la propriété Source (table ou requête)

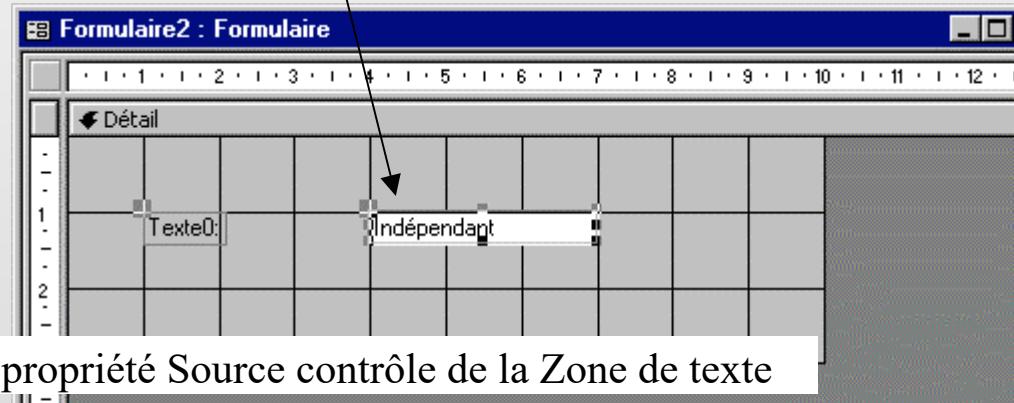
I | Les formulaires

2) Insérer dans le formulaire les Zones de texte liées aux champs de la Source de données



a) Sélectionner l'outil Zone de texte

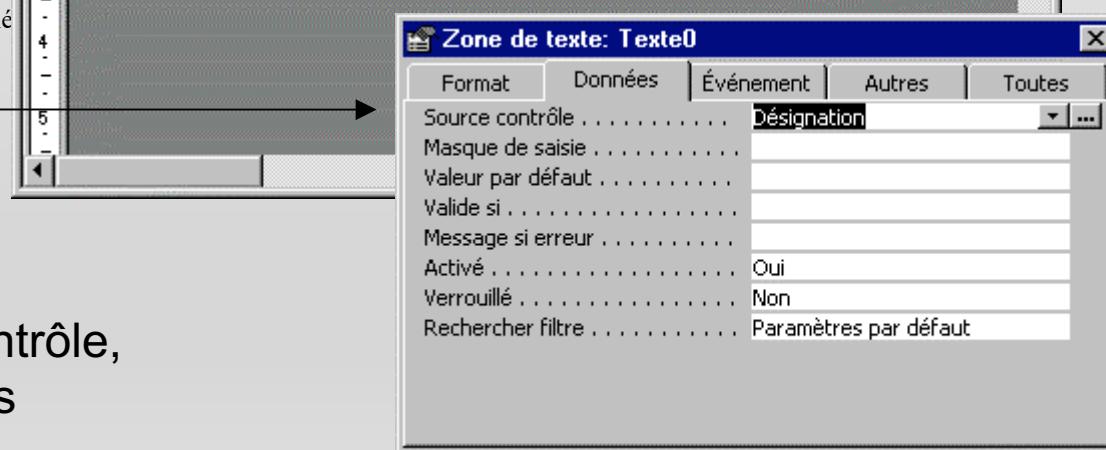
b) Insérer la Zone de texte avec son Etiquette associée



c) Définir la propriété Source contrôle de la Zone de texte

- Définir la propriété

10/30/24



Pour afficher la fenêtre des propriétés d'un contrôle, cliquer dessus avec le bouton droit de la souris

II Les états

Un état permet d'imprimer des enregistrements, en les groupant et en effectuant des totaux et des sous totaux.

En-tête d'état → *Etat du Stock*

En-tête de page → Entreprise MICRO

En-tête de groupe → Catégorie : O

Détail	IdPro	Désignation	Marque	Qstock
	10	Ps	Ibm	10
	20	Imac	Apple	20
	30	Aptiva	Ibm	10

Pied de groupe → Sous totaux : 40

...

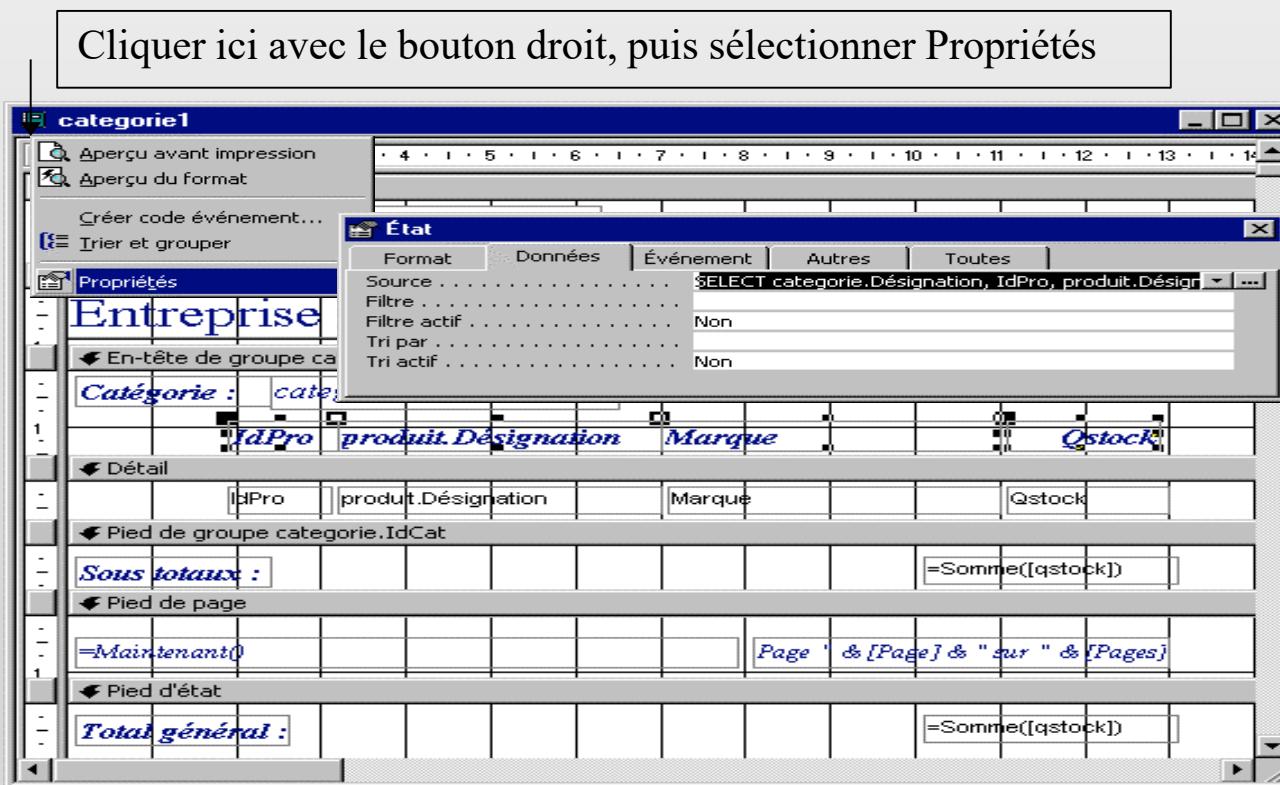
Pied de page → Jeudi 12 février 1998 Page 1 sur 1

Pied d'état → Total général : 200

II Les états

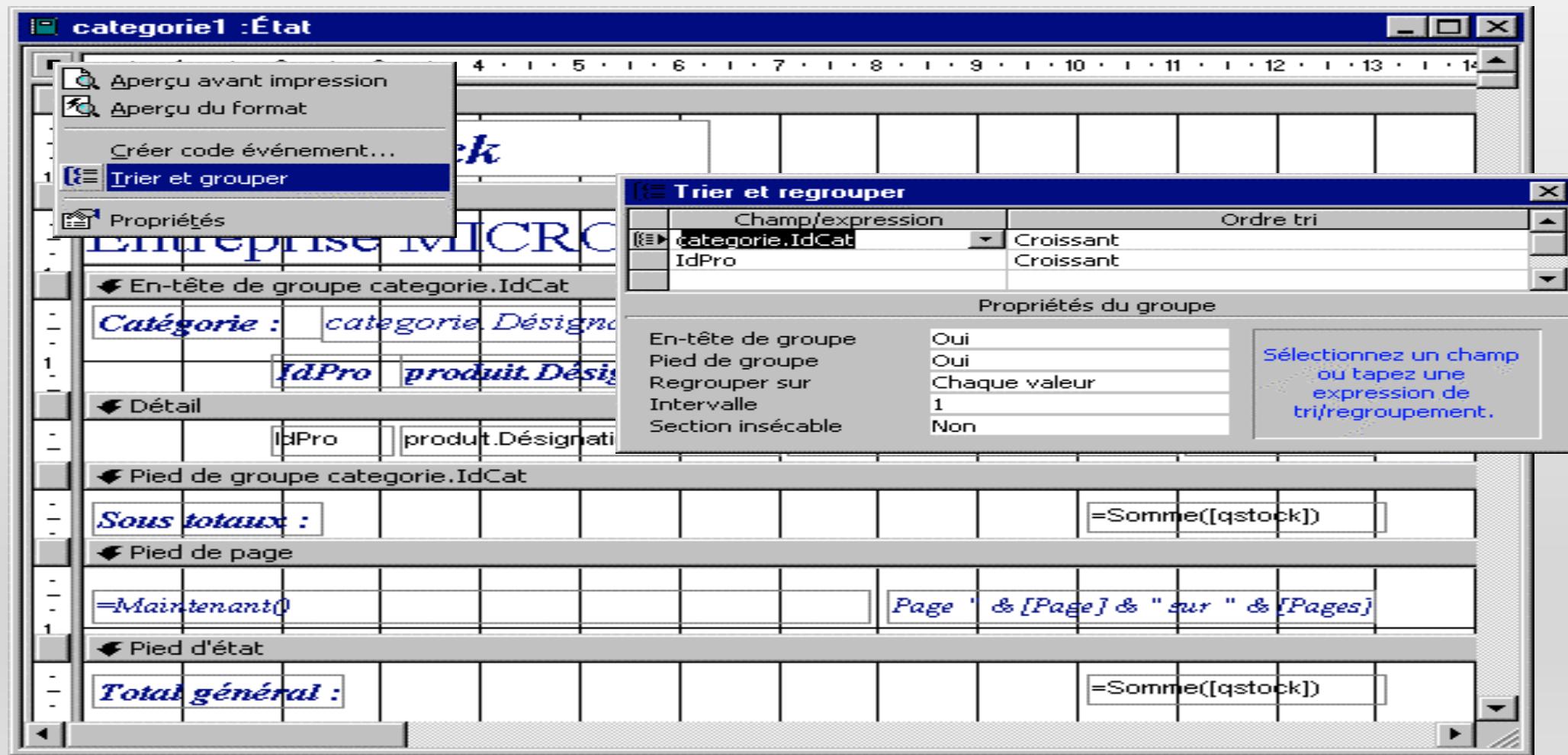
Création d'un état

- 1) Définir la propriété Source de données (table ou requête)



II Les états

2) Définir Trier et grouper



II Les états

3) Placer les champs dans les différentes section de l'état

categorie1 : État

En-tête d'état

Etat du Stock

En-tête de page

Entreprise MICRO

En-tête de groupe categorie.IdCat

Catégorie : [categorie.Désignation]

IdPro *produit.Désignation* *Marque* *Qstock*

Détail

IdPro *produit.Désignation* *Marque* *Qstock*

Pied de groupe categorie.IdCat

Sous totaux : =Somme([qstock])

Pied de page

=Maintenant() *Page ' & [Page] & " sur " & [Pages]*

Pied d'état

Total général : =Somme([qstock])

Chapitre 4 L'algèbre relationnelle

I. Les opérations

II. Le langage algébrique

I Les opérations

L'Algèbre relationnelle est une collection d'opérations

▫ OPÉRATIONS

- opérandes : 1 ou 2 relations
- résultat : une relation

▫ DEUX TYPES D'OPÉRATIONS

→ OPÉRATIONS ENSEMBLISTES

UNION
INTERSECTION
DIFFÉRENCE

→ OPÉRATIONS SPÉCIFIQUES

PROJECTION
RESTRICTION
JOINTURE
DIVISION

I Les opérations

□ UNION

L'union de deux relations R1 et R2 de même schéma est une relation R3 de schéma identique qui a pour n-uplets les n-uplets de R1 et/ou R2

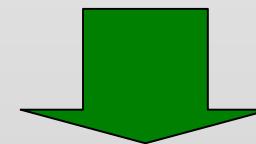
On notera :

$$R3 = R1 \cup R2$$

A	B
0	1
2	3

A	B
0	1
4	5

$$R3 = R1 \cup R2$$



A	B
0	1
2	3
4	5

I Les opérations

□ INTERSECTION

L'intersection entre deux relations R1 et R2 de même schéma est une relation R3 de schéma identique ayant pour n-uplets les n-uplets communs à R1 et R2

On notera :

$$R3 = R1 \cap R2$$

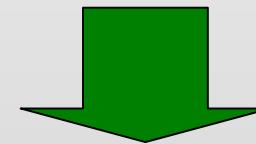
$$R1 \cap R2$$

A	B
0	1
2	3

A	B
0	1
4	5

∩

$$R3 = R1 \cap R2$$


$$R3$$

A	B
0	1

I Les opérations

□ DIFFÉRENCE

La différence entre deux relations R1 et R2 de même schéma est une relation R3 de schéma identique ayant pour n-uplets les n-uplets de R1 n'appartenant pas à R2

On notera :

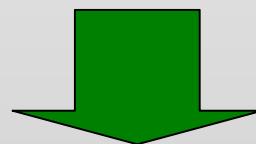
$$R3 = R1 - R2$$

R1	
A	B
0	1
2	3

-

R2	
A	B
0	1
4	5

$$R3 = R1 - R2$$



R3	
A	B
2	3

I Les opérations

□ PROJECTION

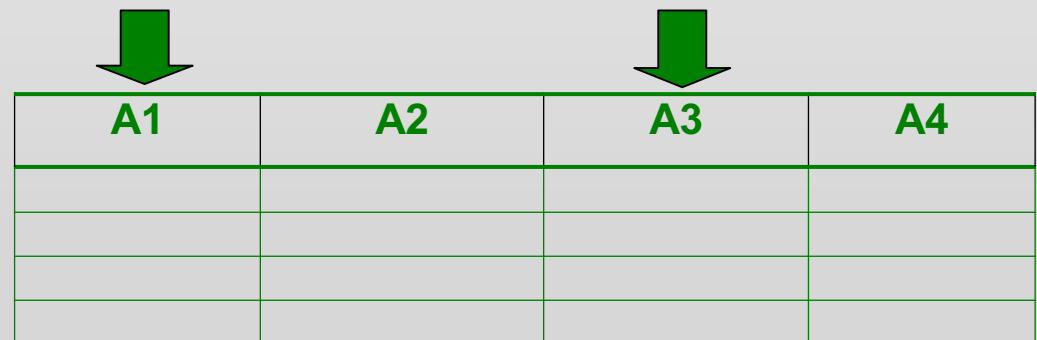
La projection d'une relation R1 est la relation R2 obtenue en supprimant les attributs de R1 non mentionnés puis en éliminant éventuellement les n-uplets identiques

On notera :

$$R2 = \pi_{A_i, A_j, \dots, A_m} (R1)$$

la projection d'une relation R1 sur les attributs A_i, A_j, ..., A_m

- La projection permet d'éliminer des attributs d'une relation
- Elle correspond à un découpage vertical :



A1	A2	A3	A4

I Les opérations

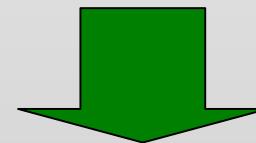
Requête 1 :

« Quels sont les références et les prix des produits ? »

PRODUIT (IdPro, Nom, Marque, Prix)

IdPro	Nom	Marque	Prix
P	PS1	IBM	1000
Q	Mac	Apple	2000
R	PS2	IBM	3000
S	Word	Microsoft	4000

$\pi_{\text{PRODUIT}} (\text{IdPro}, \text{Prix})$



IdPro	Prix
P	1000
Q	2000
R	3000
S	4000

I Les opérations

Requête 2 :

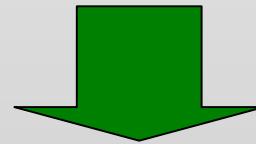
« Quelles sont les marques des produits ? »

PRODUIT (IdPro, Nom, Marque, Prix)



IdPro	Nom	Marque	Prix
P	PS1	IBM	1000
Q	Mac	Apple	2000
R	PS2	IBM	3000
S	Word	Microsoft	4000

$\pi_{\text{PRODUIT}}(\text{Marque})$



Marque
IBM
Apple
Microsoft

I Les opérations

□ RESTRICTION

La restriction d'une relation R1 est une relation R2 de même schéma n'ayant que les n-uplets de R1 répondant à la condition énoncée

→ La restriction permet d'extraire les n-uplets qui satisfont une condition

On notera :

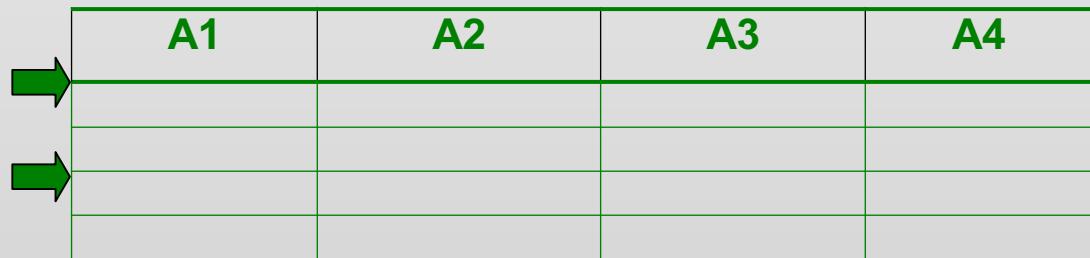
$$R2 = \sigma_{\text{condition}}(R1)$$

la restriction d'une relation R1 suivant le critère "condition"

où "condition" est une relation d'égalité ou d'inégalité entre 2 attributs ou entre un attribut et une valeur

- Elle correspond à un découpage horizontal :

A1	A2	A3	A4



I Les opérations

Requête 3 :

« Quelles sont les produits de marque 'IBM' ? »

PRODUIT (IdPro, Nom, Marque, Prix)

IdPro	Nom	Marque	Prix
P	PS1	IBM	1000
Q	Mac	Apple	2000
R	PS2	IBM	3000
S	Word	Microsoft	4000

$\sigma_{\text{PRODUIT} (\text{Marque} = \text{'IBM'})}$



IdPro	Nom	Marque	Prix
P	PS1	IBM	1000
R	PS2	IBM	3000

I Les opérations

JOINTURE

La jointure de deux relations R1 et R2 est une relation R3 dont les n-uplets sont obtenus en concaténant les n-uplets de R1 avec ceux de R2 et en ne gardant que ceux qui vérifient la condition de liaison

On notera :

$$\mathbf{R3 = R1 \times R2 \text{ (condition)}}$$

la jointure de R1 avec R2 suivant le critère condition

- Le schéma de la relation résultat de la jointure est la concaténation des schémas des opérandes (s'il y a des attributs de même nom, il faut les renommer)
- Les n-uplets de $R1 \times R2$ (condition) sont tous les couples $(u1, u2)$ d'un n-uplet de R1 avec un n-uplet de R2 qui satisfont "condition"
- La jointure de deux relations R1 et R2 est le produit cartésien des deux relations suivi d'une restriction
 - La condition de liaison doit être du type :
$$<\text{attribut1}> :: <\text{attribut2}>$$
où : attribut1 ∈ 1ère relation et attribut2 ∈ 2ème relation
:: est un opérateur de comparaison (égalité ou inégalité)

I Les opérations

- La jointure permet de composer 2 relations à l'aide d'un critère de liaison

R1(A, B, C)

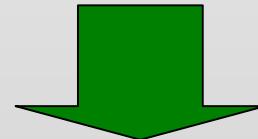
A	B	C
A1	B1	10
A2	B2	10
A3	B3	20
A4	B4	30

R2(U, V)

U	V
10	V1
20	V2
30	V3



R1 × R2 (R1.C = R2.U)



A	B	C	U	V
A1	B1	10	10	V1
A1	B2	10	10	V1
A3	B3	20	20	V2
A4	B4	30	30	V3

I Les opérations

Jointure naturelle

Jointure où l'opérateur de comparaison est l'égalité
dans le résultat on fusionne les 2 colonnes dont les valeurs sont égales

→ La jointure permet d'enrichir une relation

Requête 5 :

« Donnez pour chaque vente la référence du produit, sa désignation, son prix, le numéro de client, la date et la quantité vendue »

VENTE As V

IdCli	IdPro	Date	Qte
X	P	1/1/98	1
Y	Q	2/1/98	1
Z	P	3/1/98	1

PRODUIT As P

IdPro	Désignation	Prix
P	PS	100
Q	Mac	100



VENTE × PRODUIT (V.IdPro=P.IdPro)



Idcli	IdPro	Date	Qte	Désignation	Prix
X	P	1/1/98	1	PS	100
Y	Q	2/1/98	1	Mac	100
Z	P	3/1/98	1	PS	100

I Les opérations

□ Auto-jointure

jointure d'une relation par elle-même

Requête 6 :

« Quels sont les noms des clients qui habitent la même ville que John ?»

CLIENT As C1

IdCli	Nom	Ville
X	Smith	Nice
Y	Blake	Paris
Z	John	Nice

CLIENT As C2

IdCli	Nom	Ville
X	Smith	Nice
Y	Blake	Paris
Z	John	Nice

R1 = CLIENT × CLIENT (C1.Ville = C2.Ville)



R1

C1.IdCli	C1.Nom	Ville	C2.IdCli	C2.Nom
X	Smith	Nice	X	Smith
X	Smith	Nice	Z	John
Y	Blake	Paris	Y	Blake
Z	John	Nice	X	Smith
Z	John	Nice	Z	John

I Les opérations

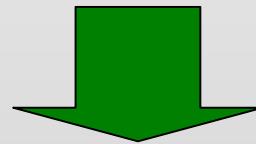
R1

C1.IdCli	C1.Nom	Ville	C2.IdCli	C2.Nom
X	Smith	Nice	X	Smith
X	Smith	Nice	Z	John
Y	Blake	Paris	Y	Blake
Z	John	Nice	X	Smith
Z	John	Nice	Z	John

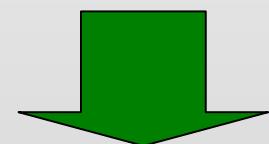
R2

C1.IdCli	C1.Nom	Ville	C2.IdCli	C2.Nom
X	Smith	Nice	Z	John
Z	John	Nice	Z	John

$R2 = \sigma_{C2.Nom} (R1)$



$R3 = \pi_{C1.Nom} (R2)$



R3

C1.Nom
Smith
John

- La normalisation conduit à décomposer ; la jointure permet de recomposer

I Les opérations

□ DIVISION

Soit deux relations

$R1 (A1, A2, \dots, An, B1, B2, \dots, Bm)$ $R2 (B1, B2, \dots, Bm)$

Si le schéma de R2 est un sous-schéma de R1. La division de R1 par R2 est une relation R3 dont

:

- le schéma est le sous-schéma complémentaire de R2 par rapport à R1
- un n-uplet (a_1, a_2, \dots, a_n) appartient à R3 si $(a_1, a_2, \dots, a_n, b_1, b_2, \dots, b_m)$ appartient à R1 pour tous $(b_1, b_2, \dots, b_m) \in R2$.

On notera :

$$R3 = R1 \div R2$$

la division de R1 par R2

I Les opérations

→ la division permet de rechercher dans une relation les sous n-uplets qui sont complétés par tous ceux d'une autre relation

Elle permet de répondre à des questions qui sont formulées avec le quantificateur universel :
"pour tout ..."

Requête 6 :

« Quels sont les élèves qui sont inscrits à tous les sports ? »

INSCRIPT

Elève	Sport
toto	judo
tata	danse
toto	foot
toto	danse

SPORT

Sport
judo
foot
danse

÷

$$\text{RES} = \text{INSCRIPT} \div \text{SPORT}$$



RES

Elève
toto

II Le langage algébrique

Le langage algébrique permet de formuler une question par une suite d'opérations de l'algèbre relationnelle

Requêtes sur le schéma CLIENT, PRODUIT, VENTE

CLIENT (**IdCli**, nom, ville)

PRODUIT (**IdPro**, désignation, marque, prix)

VENTE (**IdCli**, **IdPro**, **date**, qte)

Requête 8 :

« Donner les no des produits de marque Apple et de prix < 5000 F »

R1 = $\sigma_{\text{marque} = \text{Apple}'}(\text{PRODUIT})$

R2 = $\sigma_{\text{prix} < 5000}(\text{PRODUIT})$

R3 = R1 \cap R2

RESUL = $\pi_{\text{IdPro}}(\text{R3})$

II Le langage algébrique

Requête 9 :

« Donner les no des clients ayant acheté un produit de marque Apple »

R1 = $\sigma_{\text{PRODUIT}}(\text{marque} = \text{'Apple'})$

R2 = $R1 \times \text{VENTE}$ ($R1.\text{IdPro} = \text{VENTE.IdPro}$) RESUL =

$\pi_{R2}(\text{IdCli})$

II Le langage algébrique

Requête 10 :

« Donner les no des clients n'ayant acheté que des produits de marque Apple »

R1 = VENTE \times PRODUIT (VENTE.IdPro = PRODUIT.IdPro)

R2 = σ R1 (marque = 'Apple') R3 = π R2 (IdCli)

R4 = σ R1 (marque \neq 'Apple') R5 = π R4 (IdCli)

RESUL = R3 – R5

II Le langage algébrique

Requête 11 :

« Donner les no des clients ayant acheté tous les produits de marque Apple »

R1 = $\sigma_{\text{PRODUIT}}(\text{marque} = \text{'Apple'})$ R2 = $\pi_{R1}(\text{IdPro})$

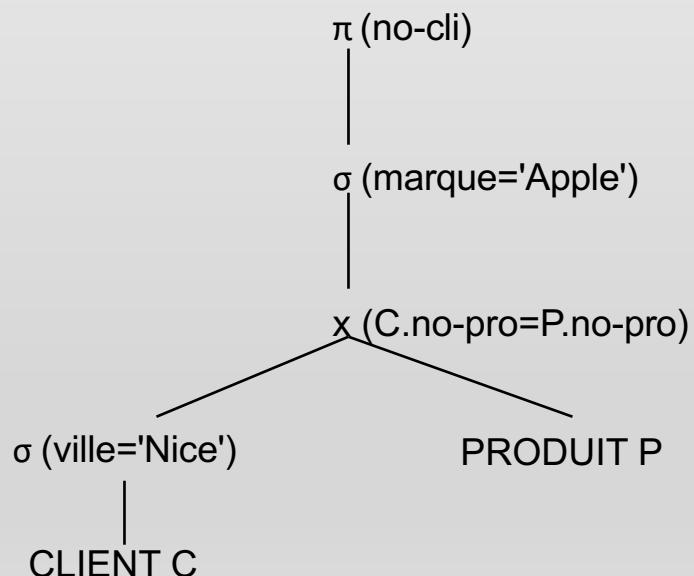
R3 = $\pi_{\text{VENTE}}(\text{IdCli}, \text{IdPro})$ R4 = $R3 \div R2$

II Le langage algébrique

□ Arbre algébrique

une question peut être représentée par un arbre

« Quels sont les clients de Nice ayant acheté un produit de marque 'Apple' ? »



II Le langage algébrique

■ Optimisation de requêtes

Plusieurs arbres équivalents peuvent être déduits d'un arbre donné à l'aide de règles de transformation simples, telles que permutation des jointures et restrictions, permutation des projections et des jointures, etc.

Ces transformations sont à la base des techniques **d'optimisation de requêtes**

Chapitre 5: Les langages de requête, QBE, SQL

I Les langages associés au modèle relationnel

II QBE (Query By Example)

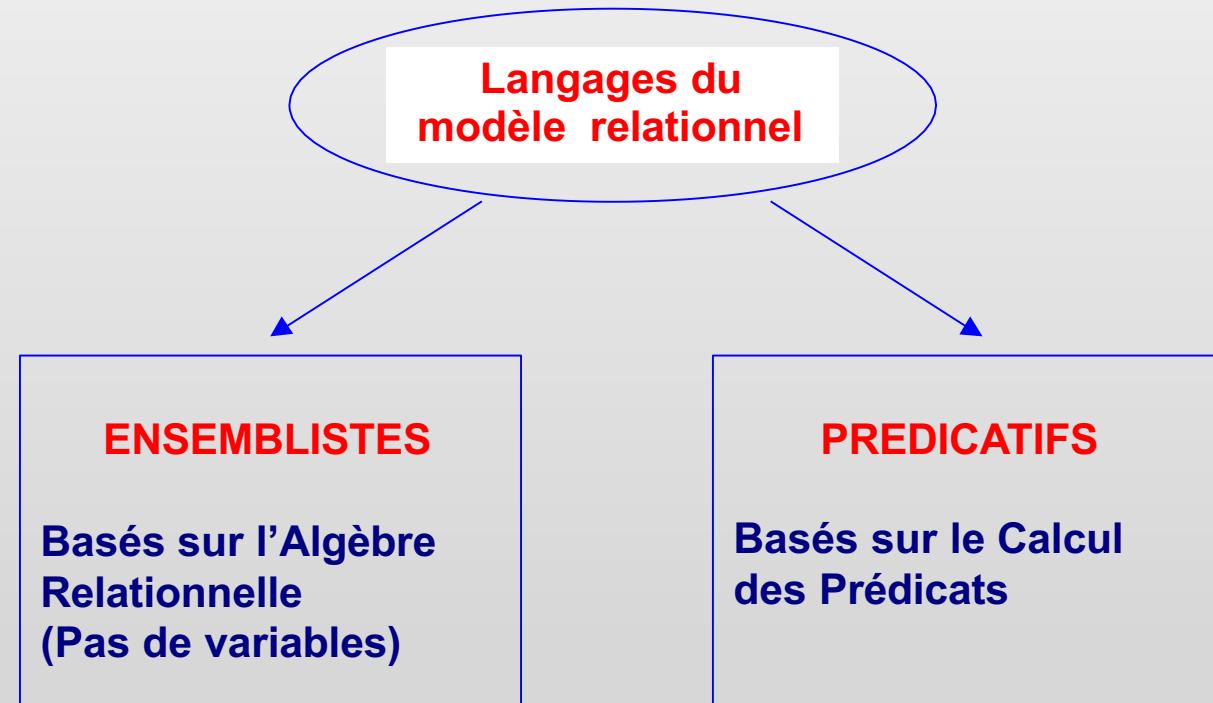
III SQL (Structured Query Language)

I Les langages associés au modèle relationnel

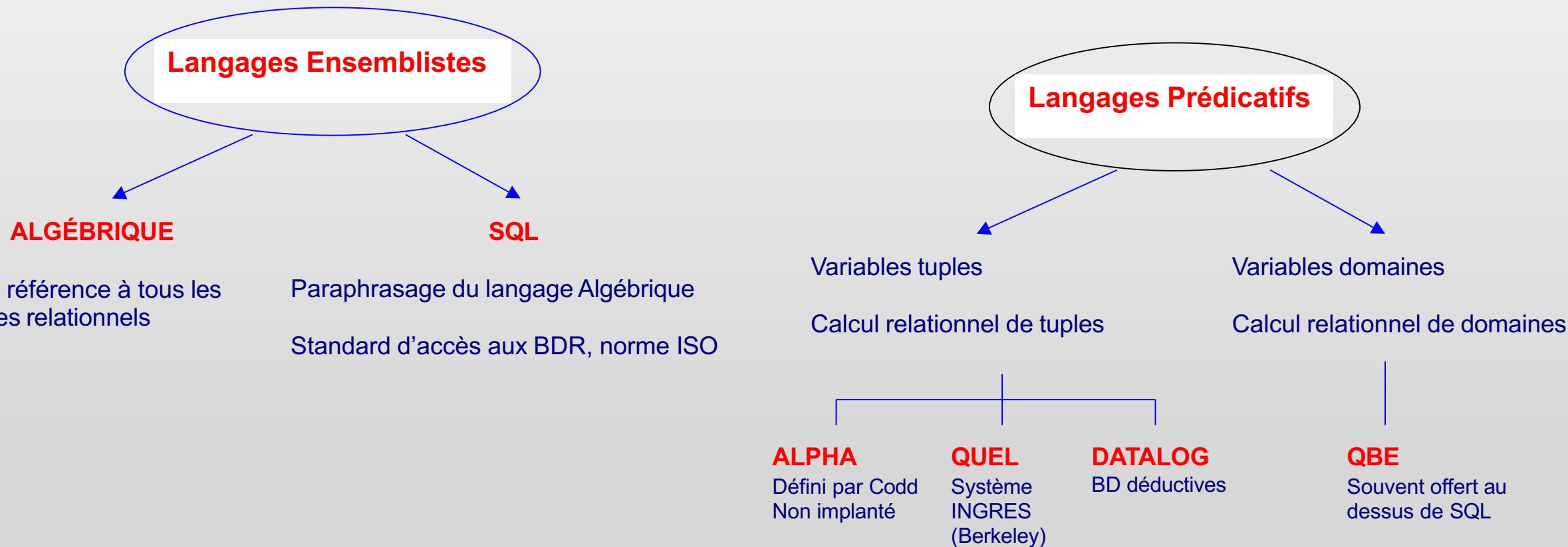
- Langages assertionnels pour décrire et manipuler les BD relationnelles

Ils permettent de spécifier les ensembles de données à sélectionner ou à mettre à jour à partir de propriétés des valeurs (*qualification*), sans dire comment retrouver les données : « *dire QUOI sans dire COMMENT* »

- Deux classes de langages correspondant à la manière de considérer une relation : comme un ensemble, ou comme un prédicat



I Les langages associés au modèle relationnel



I Les langages associés au modèle relationnel

- Exemple d'une requête dans les 4 langages ALGÉBRIQUE, SQL, ALPHA, et QBE

Donner les noms des clients qui ont acheté le produit 'p1' ?

ALGÉBRIQUE

R1 = CLIENT×VENTE (CLIENT.IdCli=VENTE.IdCli)
R2 = $\sigma_{R1} (IdPro = 'p1')$
R3 = $\pi_{R2} (\text{Nom})$

SQL

```
SELECT C.Nom  
FROM client C, vente V  
WHERE C.IdCli = V.IdCli  
AND V.IdPro = 'p1'
```

ALPHA

RANGE OF v IS vente
GET client.Nom :
 $\exists v (v.IdCli = \text{client.IdCli} \wedge v.IdPro = 'p1')$

-- v est une variable tuple
-- qualification

QBE

CLIENT	IdCli	Nom	Ville	
	X	P.		
VENTE	IdCli	IdPro	Date	Qte
	X	p1		

x est une variable domaine (*valeur exemple*)

II QBE (Query By Example)

- QBE a été développé par IBM (Zloof 77)
- L'idée de base de QBE est de formuler une question par un exemple de réponse dans les emplacements appropriés d'une table vide

Ex.: Quels sont les noms des clients de Nice ?

CLIENT	IdCli	nom	ville
		P. <u>toto</u>	Nice

P. signifie "print" ; il indique le but de la question

toto est une *valeur exemple*, i.e. un exemple de réponse possible (les valeurs exemples sont soulignées)

Nice (non souligné) est une constante

- Les variables domaines sont désignées par des **valeurs exemples soulignées**
elles servent pour établir des liens entre des lignes
si aucun lien n'est nécessaire, comme dans l'exemple ci-dessus, on peut omettre la valeur exemple (P.toto peut être réduit à P.)
- Une variable non imprimée non précédée de P. est implicitement quantifiée par "il existe"
- Une variable peut être quantifiée par "quel-que-soit" en tapant ALL. devant son nom
- La disjonction (ou) est exprimée en utilisant 2 lignes (2 exemples)

II QBE (Query By Example)

1 Opérations de recherche

Recherche simple

Q1 Donner les nos des produits vendus

VENTE		IdCli		IdPro		date		qte	
				P.p					

QBE élimine automatiquement les doublons
pour obtenir les doublons il faut spécifier ALL.

VENTE		IdCli		IdPro		date		qte	
				P.ALL.p					

Q2 Donner tous les renseignements sur tous les clients

CLIENT		IdCli		nom		ville	
		P.x		P.toto		P.rome	

on peut abréger en plaçant P. en tête

CLIENT	P.		IdCli		nom		ville	

II QBE (Query By Example)

1 Opérations de recherche

▫ Recherche qualifiée

Q3 Donner les nos des produits de marque Apple et coûtant moins de 8000 F.

PRODUIT	IdPro	nom	marque	prix
	P.p		Apple	<8000

Q5 Donner les nos des clients qui ont acheté à la fois le produit p1 et le produit p2

VENTE	IdCli	IdPro	date	qte
	P.x	p1		

Q4 Donner les noms des clients de Nice ou Rome

CLIENT	IdCli	nom	ville
	P.x		Nice
	P.y		Rome

▫ Recherche avec tri

Q6 Donner les noms des clients triés par villes et noms

CLIENT	IdCli	nom	ville
		P.AO(2).toto	P.AO(1).nice

- Le ou est exprimé en utilisant 2 lignes (2 exemples)

il faut spécifier AO. ou DO. pour obtenir des résultats triés par ordre croissant ou décroissant ; on peut préciser le champ majeur lorsque le tri s'effectue sur plusieurs champs

II QBE (Query By Example)

1 Opérations de recherche

□ Recherche utilisant un lien

Q7 Donner les noms des clients qui ont acheté le produit p1

CLIENT	IdCli	nom	ville	
	x	P. <u>toto</u>		

VENTE	IdCli	IdPro	date	qte	
	x	p1			

- la valeur exemple **x** est utilisée comme un lien entre CLIENT et VENTE

□ Recherche utilisant plusieurs liens

Q8 Donner les noms des clients qui ont acheté au moins un produit de marque 'Apple'

CLIENT	IdCli	nom	ville	
	x	P. <u>toto</u>		

PRODUIT	IdPro	nom	marque	prix	
	p		Apple		

VENTE	IdCli	IdPro	date	qte	
	x	p			

II QBE (Query By Example)

1 Opérations de recherche

□ Recherche utilisant la négation

Q9 Donner les noms des clients qui n'ont pas acheté le produit p1

CLIENT		IdCli		nom		ville	
		x		P. <u>toto</u>			

VENTE		IdCli		IdPro		date		qte	
-		x		p1					

- Notez l'opérateur négation (\neg) portant sur la ligne de VENTE

□ Recherche utilisant un lien dans une seule table

Q10 Donner les nos des clients qui ont acheté au moins un produit acheté par le client c1

VENTE		IdCli		IdPro		date		qte	
		P. c1		p p					

II QBE (Query By Example)

1 Opérations de recherche

▫ Recherche utilisant une boîte condition

Q11 Donner les nos des produits achetés par plus qu'un seul client

VENTE		IdCli		IdPro		date		qte	
		x		P.	p				
		y		p					

| CONDITION BOX |
x ≠ y

- Lorsque la condition est importante ou si l'on veut simplifier l'écriture d'une requête on peut utiliser une *boîte condition*

II QBE (Query By Example)

2 Fonctions de calcul

- QBE fournit un ensemble de fonctions de calcul prédéfinies :

CNT.ALL

SUM.ALL

AVG.ALL

MAX.ALL

MIN.ALL

CNT.UNQ.ALL

SUM.UNQ.ALL

AVG.UNQ.ALL

- **Recherche simple utilisant une fonction**

Q12 Donner le nombre total de clients

CLIENT	IdCli	nom	ville	
	P.CNT.ALL.X			

- ALL. est toujours spécifié
- L'option UNQ. signifie "éliminer les doublons avant d'appliquer la fonction"
- Les valeurs nulles sont éliminées, sauf pour CNT.

Q13 Donner le nombre total de clients ayant acheté des produits

VENTE	IdCli	IdPro	date	qte	
	P.CNT.UNQ.ALL.X				

II QBE (Query By Example)

2 Fonctions de calcul

□ Recherche qualifiée utilisant une fonction

Q14 Donner le nombre de ventes du produit p1

VENTE	IdCli	IdPro	date	qte
	P.CNT.ALL. <u>x</u>	p1		

Q15 Donner la qte totale vendue du produit p1

VENTE	IdCli	IdPro	date	qte	
		p1		P.SUM.ALL. <u>qte</u>	

□ Fonction dans la boîte condition

Q16 Donner les nos des produits moins chers que tous ceux de marque Apple

PRODUIT	IdPro	nom	marque	prix	
	P. <u>p</u>		Apple	<u>val</u>	<u>prix</u>

| CONDITION BOX |
| val <MIN.ALL.prix |

II QBE (Query By Example)

2 Fonctions de calcul

▫ Recherche avec regroupement

Q17 Pour chaque produit vendu, donner le no de produit et la quantité totale vendue de ce produit

VENTE		IdCli		IdPro		date		qte	
				P.G.p				P.SUM.ALL.qte	

G. est l'opérateur de regroupement

II QBE (Query By Example)

3 Opérations de mise à jour

- Les opérateurs de mise à jour sont :

- **UPDATE** (modification)
- **INSERT** (insertion)
- **DELETE** (suppression)

Le nom des opérateurs apparaît sous le nom de la relation abrégé par U. , I. ou D.

□ Modification d'un seul enregistrement

- L'enregistrement à modifier est identifié par la valeur de sa clé primaire
- Les valeurs de clé primaire ne peuvent pas être modifiées

Q18 Changer la ville du client c1 par Nice

CLIENT	U.	IdCli	nom	ville
		c1		Nice

Q19 Augmenter de 100F le prix du produit p1

PRODUIT	U.	IdPro	nom	marque	prix
		p1			\underline{val}

II QBE (Query By Example)

3 Opérations de mise à jour

□ Modification de plusieurs enregistrements

- Les enregistrements à modifier sont spécifiés par une valeur exemple (non une constante) de clé primaire

Q20 Doubler le prix de tous les produits Apple

PRODUIT	IdPro	nom	marque	prix	
	p		Apple	val	
U.	p			2 val	

Q21 Mettre à 0 toutes les qtés achetées par les clients de Nice

CLIENT	IdCli	nom	ville	
	x		Nice	

VENTE	IdCli	IdPro	date	qte	
U.	x			0	

□ Insertion d'un nouvel enregistrement

Q22 Ajouter le client (c20, 'Duduche', 'Nice') dans la table CLIENT

CLIENT	IdCli	nom	ville
I.	c20	Duduche	Nice

- Le nouvel enregistrement doit avoir une valeur de clé primaire non nulle, et distincte de toutes les valeurs de clés primaires existantes dans la table

II QBE (Query By Example)

3 Opérations de mise à jour

□ Insertion de plusieurs enregistrements

Q23 La table TEMP a une seule colonne IdPro. Introduire dans TEMP les nos de tous les produits achetés par le client c1

TEMP	I.	IdPro
	I.	p

VENTE		IdCli		IdPro		date		qte	
		c1		p					

□ Suppression de plusieurs enregistrements

Q25 Supprimer tous les clients de Nice

CLIENT	D.	IdCli	nom	ville
	D.			Nice

- A nouveau la suppression n'est possible qu'en respect de l'*intégrité référentielle* avec VENTE

□ Suppression d'un seul enregistrement

Q24 Supprimer le client c1

CLIENT	D.	IdCli	nom	ville
	D.	c1		

- La suppression n'est pas possible si le client figure dans la table VENTE (*intégrité référentielle*)

II QBE (Query By Example)

4 Dictionnaire de données QBE

- Le **dictionnaire de données** contient la description des relations du schéma relationnel de la BD
- Le DD est représenté sous forme de relations prédéfinies, parmi lesquelles :
 - Le DD peut être interrogé et mis à jour avec le langage de manipulation de données

CATALOG (TNAME, CREATOR, NCOLS, ...)

informations sur toutes les tables de la BD

COLUMNS (TNAME, CNAME, TYPE, LENGTH, ...)

informations sur les colonnes de toutes les tables de la BD

II QBE (Query By Example)

5 Fermeture transitive

- La fermeture transitive permet d'enrichir une relation à 2 attributs de même domaine, avec tous les couples qui se déduisent par transitivité

Si on a (a, b) et (b, c) alors on ajoute (a, c)

Ex.

- La fermeture transitive ne peut pas être exprimée par une expression constante de l'algèbre relationnelle elle peut être effectuée par une série de jointure/projection/union, mais le nbre d'opérations à effectuer dépend du contenu de la relation

NOMENCLATURE	ENSEMBLE	SOUS-ENSEMBLE
<pre>graph TD; A --- B; A --- C; C --- E; C --- F; E --- G;</pre>	A A C C E	B C E F G
Fermeture transitive	A A C C E A A C	B C E F G E F G G

II QBE (Query By Example)

5 Fermeture transitive

- QBE permet d'exprimer des questions de type fermeture transitive

Q26 Donner les sous-ensembles au 2ième niveau de l'ensemble A

NOMENCLATURE	ensemble A	sous-ens <u>u</u>	P. <u>v</u>

Pour rechercher les sous-ensembles de 3ième niveau il faut ajouter une ligne :

NOMENCLATURE	ensemble A	sous-ens <u>u</u>	P. <u>v</u>

Pour rechercher les sous-ensembles de niveau n, il faut n lignes ; QBE fournit un raccourci en permettant d'indiquer le nombre de niveau :

NOMENCLATURE	ensemble A	sous-ens P.u(3L)

(3L) signifie "third level"

QBE permet aussi d'obtenir les sous-ensembles de dernier niveau à l'aide du mot clé LAST