

# Elaboration d'un Modèle Logique de Données Relationnel (MLD-R)



Bernard ESPINASSE  
Professeur à Aix-Marseille Université (AMU)  
Ecole Polytechnique Universitaire de Marseille



Novembre 2012

- Problématique du MLD
- Formalisme graphique de Merise
- Dérivation d'un MLD-R à partir d'un MCD en Entité-Relation
- Création de tables en langage SQL (clés primaires et étrangères)
- Dimensionnement d'une BD Relationnelle

## Plan

1. Problématique du MLD-R
2. Formalisme graphique de Merise
3. Dérivation d'un MLD-R à partir d'un MCD en Entité-Relation
4. Création de tables en langage SQL (clés primaires et étrangères)
5. Dimensionnement d'une BD Relationnelle (multiplicité moyenne des liens)

## Problématique du MLD

### Modèle Conceptuel de Données (MCD) :

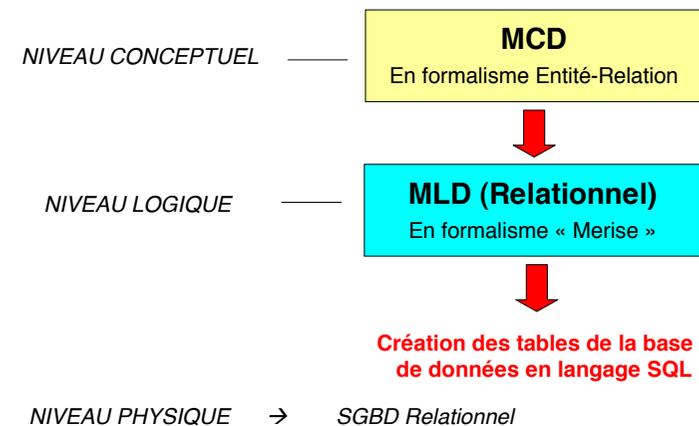
- permet de modéliser la **sémantique des informations** d'une façon **compréhensible par l'utilisateur** de la future base de données
- utilise le formalisme (graphique) **Entité-Relation**
- **ne permet pas d'implémentation informatique de la base de données** dans un SGBD donné

### Modèle Logique de Données (MLD) :

- permet de modéliser la **structure selon laquelle les données seront stockées** dans la future base de données
- est **adapté à une famille de SGBD** : ici les **SGBD relationnels** (MLD Relationnels ou **MLD-R**)
- utilise le formalisme graphique **Merise**
- permet **d'implémenter** la base de données dans un **SGBD donné**

## Démarche d'élaboration d'un MLD Relationnel

- **MCD** : *Modèle Conceptuel de Données*
- **MLD-R** : *Modèle Logique de Données Relationnel*



## Formalisme graphique "Merise" pour le MLD-R (1)

### Table

Table EMPLOYE (représentation graphique) :

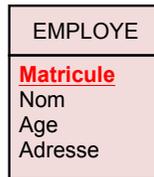


Schéma de la table EMPLOYE :

EMPLOYE (Matricule, nom, age, adresse)

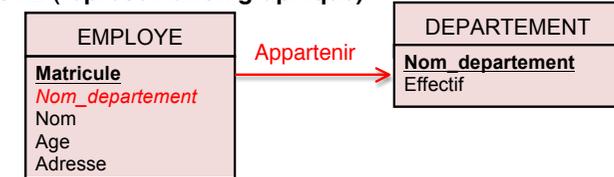
Attributs de la table EMPLOYE :

- Matricule : clé primaire
- Nom, Age, Adresse : autres attributs

## Formalisme graphique "Merise" pour le MLD-R (2)

### Lien entre tables : contrainte d'intégrité référentielle

Table EMPLOYE (représentation graphique) :

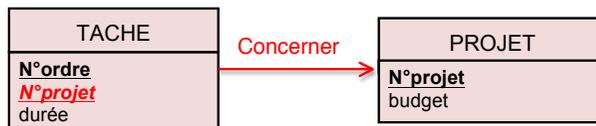


Schémas des tables :

- Table DEPARTEMENT (Nom\_departement, Effectif) :
  - Nom\_departement : clé primaire
- Table EMPLOYE (Matricule, Nom\_departement, Nom, Age, Adresse)
  - Matricule : clé primaire
  - Nom\_departement : **clé étrangère** vers table DEPARTEMENT

## Formalisme graphique "Merise" du MLD-R (3)

### Lien entre tables : clé primaire composée référentielle

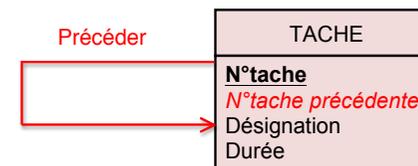


Schémas des tables :

- Table PROJET (N°projet, budget)
  - N°projet : clé primaire
- Table TACHE (N°ordre, N°projet, durée)
  - N°ordre, n°projet : **clé primaire composée**
  - N°projet : **clé étrangère** vers table PROJET

## Formalisme graphique "Merise" du MLD-R (4)

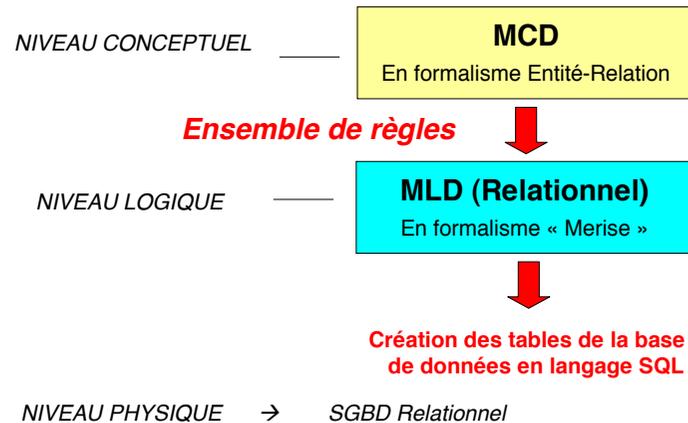
### Lien entre tables : contraintes d'intégrité référentielle réflexive



Schémas des tables :

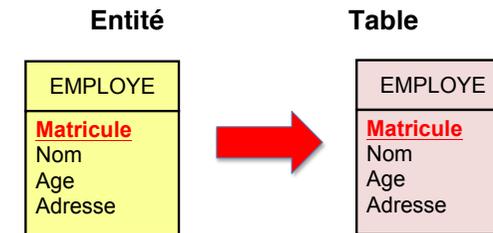
- Table TACHE (n°tache, n°tache précédente, désignation, durée)
  - N°tache : **clé primaire**
  - N°tache précédente : **clé étrangère** vers table TACHE

## Dérivation d'un MLD-R à partir d'un MCD en Entité-Relation



## MCD -> MLD : dérivation des entités

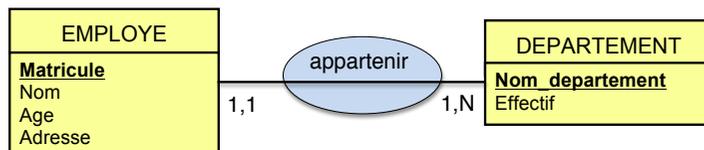
Règle : toute entité du MCD se dérive en une table du MLD



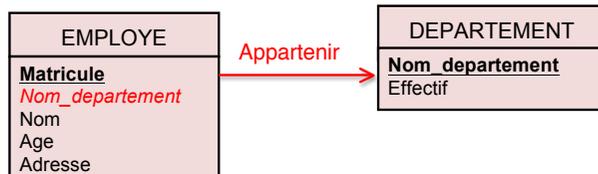
la propriété identifiante de l'entité devient la clé primaire de la table

## MCD -> MLD : relations (\*,N)-(1,1)

MCD :



MLD :

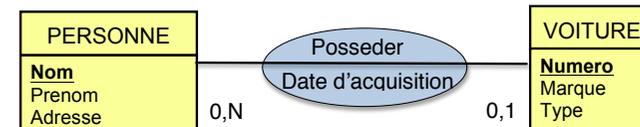


Schémas relationnels :

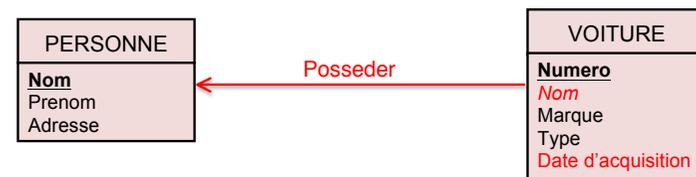
- Table DEPARTEMENT (Nom\_departement, Effectif) :
- Table EMPLOYE (Matricule, *Nom\_departement*, Nom, Age, Adresse)  
*Nom\_departement* : clé étrangère vers table DEPARTEMENT

## MCD -> MLD : relations (\*,N)-(0,1)

MCD :



MLD :

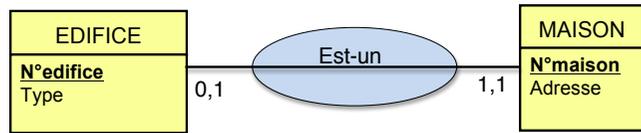


Schémas relationnels :

- PERSONNE (Nom, Prenom, Adresse) ;
- VOITURE (Numéro, *Nom*, Marque, Type, *Date\_acquisition*) ;

### MCD -> MLD : relations (0,1)-(1,1)

**MCD :**



**MLD :**

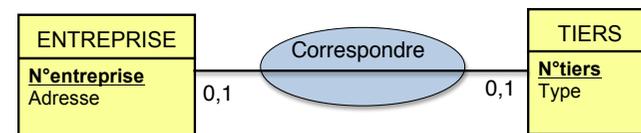


**Schémas relationnels :**

- EDIFICE (N°edifice, Type) ;
- MAISON (N°maison, N°édifice, Adresse).

### MCD -> MLD : relations (0,1)-(0,1)

**MCD :**



**MLD (solution 1) :**



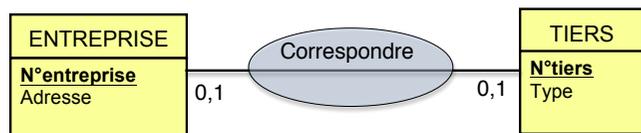
**Schémas relationnels :**

- ENTREPRISE (N°entreprise, Adresse) ;
- TIERS (N°tiers, N°entreprise, Adresse)

La cardinalité (0,1) pose le problème d'accepter des valeurs nulles sur l'attribut migrant pouvant fixer le sens de migration (par exemple la taille des clés).

### MCD -> MLD : relations (0,1)-(0,1)

**MCD :**



**MLD (solution 2) :**



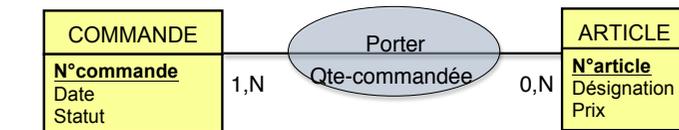
**Schémas relationnels :**

- ENTREPRISE (N°entreprise, N°tiers, Adresse) ;
- TIERS (N°tiers, Adresse)

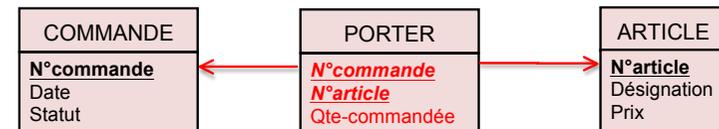
Idem : la cardinalité (0,1) pose le problème d'accepter des valeurs nulles sur l'attribut migrant pouvant fixer le sens de migration (par exemple, la taille des clés).

### MCD -> MLD : relations (0/1,N)-(0/1,N)

**MCD :**



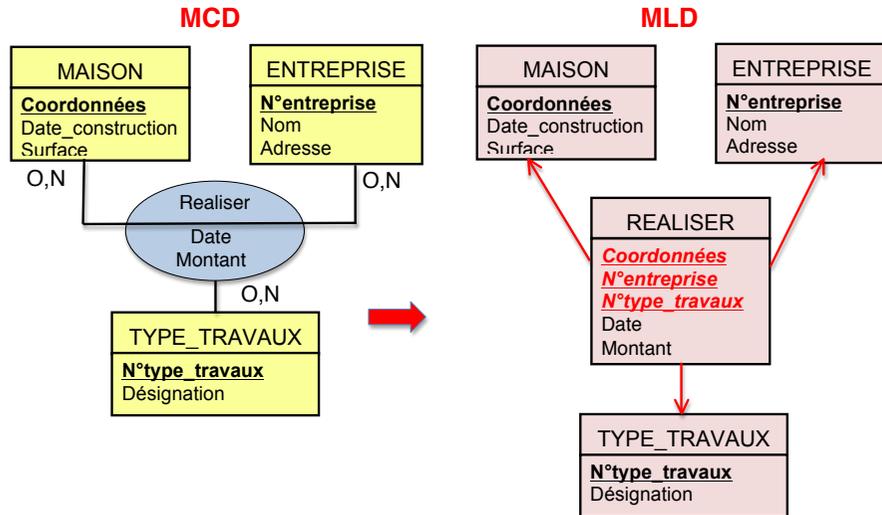
**MLD :**



**Schémas relationnels :**

- COMMANDE (N°commande, Date, Statut) ;
- PORTER (N°article, N°commande, Qte\_commandée) ;
- ARTICLE (N°article, Désignation, Prix).

## MCD -> MLD : relations ternaires ou plus (1)



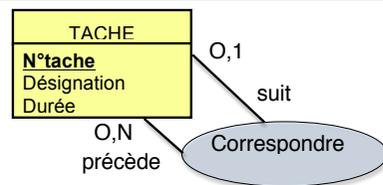
## MCD -> MLD : relations ternaires ou plus (2)

### Schémas relationnels associés :

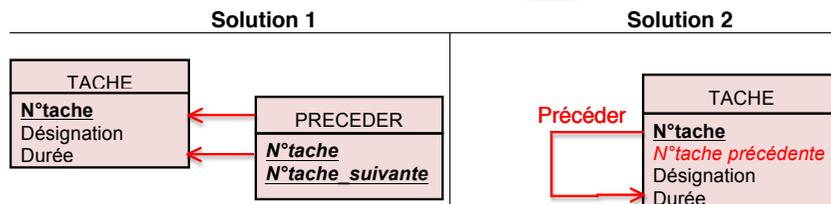
- MAISON (Coordonnées, Date\_construction, Surface) ;
- TYPE\_TRAVAUX (N°type\_travaux, Désignation) ;
- RÉALISER (N°entreprise, Coordonnées, N°type\_travaux, Date, Montant) ;
- ENTREPRISE (N°entreprise, Nom, Adresse).

## MCD -> MLD : Relations réflexives (0,N)-(0,1)

### MCD :



### MLD :



### Schémas relationnels :

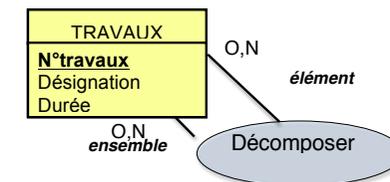
#### Solution 1 :

- TACHE (N°tache, Désignation, Durée) ;
- PRÉCÉDER (N°tache, N°tache\_suivante)

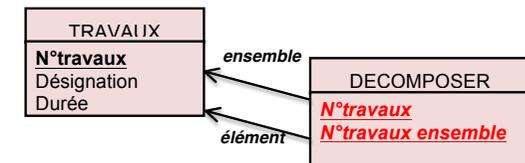
#### Solution 2 : TACHE (N°tache, N°tache\_précédente, Désignation, Durée)

## MCD -> MLD : Relations réflexives (\*,N)-(\*,N)

### Entité-Relation :



### Relationnel dérivé :

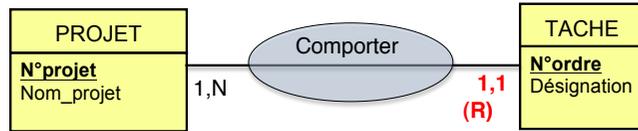


### Schémas relationnels :

- TRAVAUX (n°travaux, désignation, durée) ;
- DÉCOMPOSER (n°travaux, n°travaux\_ensemble).

## Règle de dérivation : Identifiant relatif

MCD :



MLD :

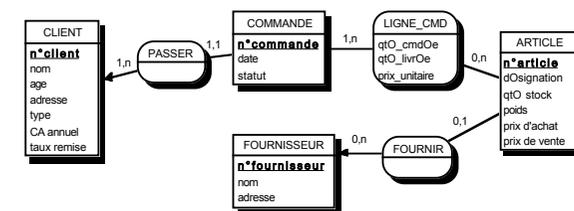


Schémas relationnels :

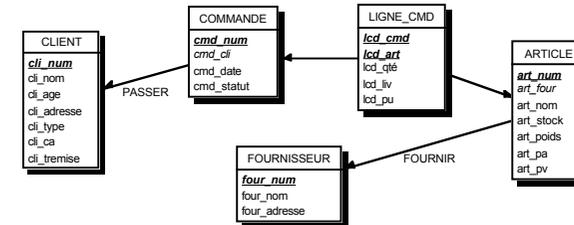
- PROJET (N°projet, Nom\_projet)
- TRANCHE (N°ordre, N°projet, Désignation)

## Exemple de passage E-R au Relationnel

Entité-Relation :

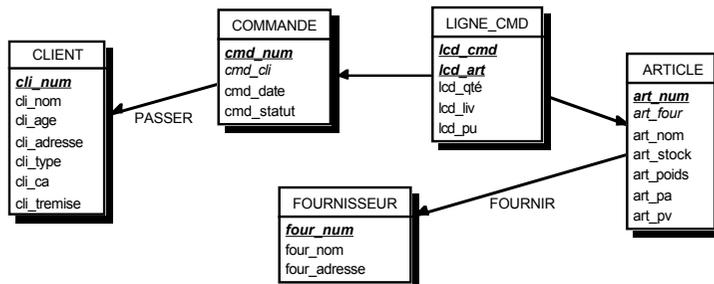


Relationnel dérivé :



## Exemple de passage E-R au relationnel

• **Modèle relationnel dérivé :**



• **Schémas relationnels :**

- CLIENT (cli\_num, cli\_nom, cli\_age, cli\_adresse, cli\_type, cli\_ca, cli\_tremise)
- ARTICLE (art\_num, art\_nom, art\_four, art\_stock, art\_poids, art\_pa, art\_pv)
- COMMANDE (cmd\_num, cmd\_cli, cmd\_date, cmd\_statut)
- LIGNE\_CMD (lcd\_cmd, lcd\_art, lcd\_qte, lcd\_liv, lcd\_pu)
- FOURNISSEUR (four\_num, four\_nom, four\_adresse)

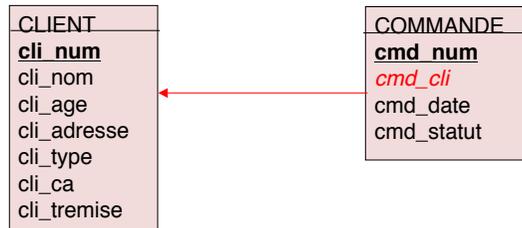
## Introduction au langage SQL

- Origine :** SQL (Structured Query Language) est un langage de requêtes standard pour les SGBD relationnels
- 3 niveaux de normes :**
  - SQL86 (standard ANSI en 86 puis ISO en 87) : la base puis SQL89 ou SQL1 : l'intégrité:
  - SQL91 ou SQL2
  - SQL3 (98) : SQL devient un langage de programmation et évolue vers l'objet

Dans ce chapitre nous ne considérerons que la création **BASIQUE** de tables par la commande **CREATE TABLE** de SQL2,

## Création d'une table en langage SQL (1)

Soit le MLD suivant :



**Schémas relationnels :**

**Table CLIENT** (cli\_num, cli\_nom, cli\_age, cli\_adresse, cli\_type, cli\_ca, cli\_tremise)

- cli\_num = clé primaire

**Table COMMANDE** (cmd\_num, cmd\_cli, cmd\_date, cmd\_statut)

- cmd\_num = clé primaire
- cmd\_cli, = clé étrangère

## Création d'une table (2) : clé primaire

**Table CLIENT** (cli\_num, cli\_nom, cli\_age, cli\_adresse, cli\_type, cli\_ca, cli\_tremise)

```
CREATE TABLE Client
(cli_num CHAR(8) NOT NULL,
 cli_nom CHAR(25) NOT NULL,
 cli_age INTEGER NOT NULL,
 cli_adresse VARCHAR(80),
 cli_type VARCHAR(16),
 cli_ca INTEGER,
 cli_tremise INTEGER NOT NULL,
 PRIMARY KEY (cli_num)) ;
```

**NOT NULL** : on n'accepte pas que l'attribut puisse avoir une valeur nulle (valeur inconnue)

**PRIMARY KEY (cli\_num)**: l'attribut cli\_num est **clé primaire** de la table Client.

**Remarque** : un attribut déclaré clé primaire doit être défini avec l'option NOT NULL

## Création d'une table (3) : clé étrangère

**Table COMMANDE** (cmd\_num, cmd\_cli, cmd\_date, cmd\_statut)

**Création de la table COMMANDE :**

```
CREATE TABLE Commande
(cmd_num CHAR(8) NOT NULL,
 cmd_cli CHAR(8) NOT NULL,
 cmd_date DATE NOT NULL,
 cmd_statut VARCHAR(16),
 PRIMARY KEY (cmd_num),
 FOREIGN KEY (cmd_cli) REFERENCES client);
```

**PRIMARY KEY (cmd\_num)**: l'attribut cmd\_num est clé primaire de la table Commande.

**FOREIGN KEY (cmd\_cli) REFERENCES client**: l'attribut cmd\_cli est une clé étrangère qui réfère à la table Client

**Remarques :**

- 1- la table client doit déjà exister
- 2- l'attribut cmd\_cli de la table commande est du même type que celui de la clé primaire de la table client.

## Création d'une table (4) : clé étrangère

**Table ARTICLE** (art\_num, art\_nom, art\_four, art\_stock, art\_poids, art\_pa, art\_pv) ;

• **Création de la table ARTICLE :**

```
CREATE TABLE Article
(art_num CHAR(8) NOT NULL,
 art_nom VARCHAR(25) NOT NULL,
 art_four CHAR(8) NOT NULL,
 art_stock INTEGER NOT NULL,
 art_poids NUMERIC (8,1),
 art_pa INTEGER NOT NULL,
 art_pv INTEGER NOT NULL,
 PRIMARY KEY (art_num),
 FOREIGN KEY (art_four) REFERENCES Fournisseur);
```

**PRIMARY KEY (art\_num)**: l'attribut art\_num est clé primaire de la table article.

**FOREIGN KEY (art\_four) REFERENCES Fournisseur**: l'attribut art\_cli est une clé étrangère qui réfère à la table Fournisseur.

## Création d'une table (5) : clé primaire composée

Table **LIGNE\_CMD** (*lcd\_cmd, lcd\_art, lcd\_qte, lcd\_liv, lcd\_pu*)

```
CREATE TABLE Ligne_cmd
(lcd_art CHAR(8) NOT NULL,
lcd_cmd INTEGER NOT NULL,
lcd_qte INTEGER NOT NULL,
lcd_liv INTEGER,
lcd_pu INTEGER NOT NULL,
FOREIGN KEY (lcd_cmd) REFERENCES Commande,
FOREIGN KEY (lcd_art) REFERENCES Article,
PRIMARY KEY (lcd_cmd, lcd_art)) ; <- Clé primaire composée
```

**FOREIGN KEY (lcd\_cmd) REFERENCES commande:** l'attribut lcd\_cmd est une clé étrangère qui réfère à la table Commande.

**FOREIGN KEY (lcd\_art) REFERENCES commande:** l'attribut lcd\_art est une clé étrangère qui réfère à la table Article.

**PRIMARY KEY (lcd\_cmd, lcd\_art):** la clé primaire de la table Ligne\_cmd est composée des attributs lcd\_cmd et lcd\_art qui sont ici clés étrangères.

## Insertion d'enregistrement dans une table en SQL

Table **CLIENT** (*cli\_num, cli\_nom, cli\_age, cli\_adresse, cli\_type, cli\_ca, cli\_tremise*)

```
CREATE TABLE Client
(cli_num CHAR(8) NOT NULL,
cli_nom CHAR(25) NOT NULL,
cli_age INTEGER NOT NULL,
cli_adresse VARCHAR(80),
cli_type VARCHAR(16),
cli_ca INTEGER,
cli_tremise INTEGER NOT NULL,
PRIMARY KEY (cli_num)) ;
```

Insertion d'un nouveau enregistrement dans la table Client :

```
INSERT INTO client VALUES ('C2345', 'Tranvouez', 29,
'Marseille', 'particulier', 3680, 25) ;
```

## Suppression d'enregistrement dans une table

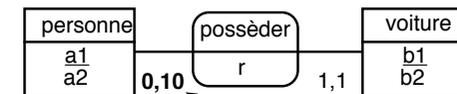
Table **CLIENT** (*cli\_num, cli\_nom, cli\_age, cli\_adresse, cli\_type, cli\_ca, cli\_tremise*) ;

```
CREATE TABLE Client
(cli_num CHAR(8) NOT NULL,
cli_nom CHAR(25) NOT NULL,
cli_age INTEGER NOT NULL,
cli_adresse VARCHAR(80),
cli_type VARCHAR(16),
cli_ca INTEGER,
cli_tremise INTEGER NOT NULL,
PRIMARY KEY (cli_num)) ;
```

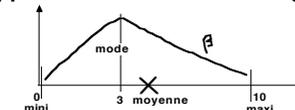
Suppression d'un enregistrement dans la table client :

```
DELETE FROM client WHERE (cli_nom = 'Tranvouez');
```

## Dimensionnement d'une BD Relationnelle : Multiplicités moyennes des liens relationnels



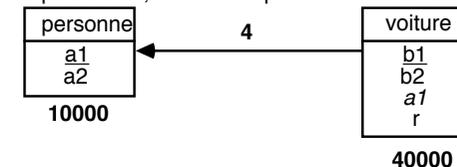
supposons une distribution triangulaire:



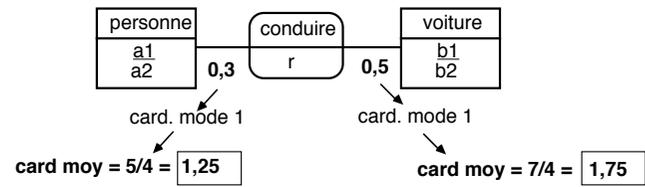
selon la distribution, ici une loi triangulaire d'où moy = (mini+2(mod)+max)/4

d'où **cardinalité moyenne** = (0+2(3)+10)/4 = 4

si 10000 personnes, la relation "posséder" aura 40000 tuples



## Propagation des multiplicités moyenne



d'où:

nb personne x 1,25 = nb voiture x 1,75 soit:

