# A. Modèle Ensembliste

## 1. Modèle ensembliste



Pour leur conception, les bases de données sont ici vues comme des ensembles constitués de plusieurs populations d'objets *en interaction*, participant au bon fonctionnement d'un certain *système*. Établir un schéma de base de données consiste à décrire ces différentes populations d'objets, mais surtout et principalement à décrire les dépendances et les interactions entre ces populations.

Une base de donnée est constituée de plusieurs ensembles d'objets et d'opérateurs participant au bon fonctionnement d'un système:

Exemple 1 :

- Ensembles d'employés
- Ensembles de commandes
- Ensembles d'articles
- Ensembles de clients

Exemple 2 :

- Ensembles d'étudiants
- Ensembles de séances
- Ensembles de cours
- Ensembles de copies

On parle plus généralement d'**ensembles d'entités**.



** Le modèle entité/association **

Le modèle entité/associations est une méthode de description des relations entre ensembles d’entités. Il s’appuie sur le prédicat selon lequel tous les éléments des ensembles d’entités sont discernables.

Le modèle entités/associations repose sur un langage graphique de description des données, indépendant du support et de la mise en œuvre informatique.

#### Généralités

Une **entité** $x$

- est une représentation d'un objet du monde réel,
- appartenant au système/à l'organisation modélisée.
- Une entité est décrite par une ou plusieurs valeurs caractéristiques, appelées **attributs**.

Les informations conservées au sujet des entités d'un ensemble sont les **attributs**.

- Chaque **attribut** :
  - a un **nom** unique dans le contexte de cet ensemble d'entités : $A$, $B$, $C$, $A_1$, $A_2$, ..., $A_m$, ...
    - Exemples de noms concrets : *couleur*, *nom*, *horaire*, *salaire*.
  - prend ses valeurs dans un domaine bien spécifié,
    - également appelé le **type** de l'attribut.
    - Le domaine d'un attribut est noté $\text{dom}(Aj)= Dj$.
      - Exemples :
        - $\text{dom}(couleur)={rouge, vert, bleu, jaune}$,
        - $\text{dom}(nom) = $ensemble des chaînes de caractères,
        - $\text{dom}(salaire) =$ entiers naturels
        - etc...



- Un attribut $A_j$ est une fonction à valeur sur $D_j$ :

$$A_j : E \rightarrow D_j$$ $$x \mapsto A_j(x)$$



- Un attribut peut être :
  - simple ou composé.
    - Exemple : une *adresse* peut être décrite par une simple chaîne de caractères, ou peut être décomposée en *rue* , *no*, *boîte*, *ville*, *code postal*, *pays*.
  - obligatoire ou facultatif ($D_j$ peut ou non contenir la valeur ø ).
  - atomique ou non (Un attribut peut posséder 0, 1 voire plusieurs valeurs...)

Un **ensemble d'entités** est un ensemble fini d'éléments : $$E = \{x_1,...,x_n\}$$ Il regroupe (ou associe) plusieurs entités ayant des caractéristiques communes (descriptibles à l'aide du même ensemble d'attributs).



**Exemples** :

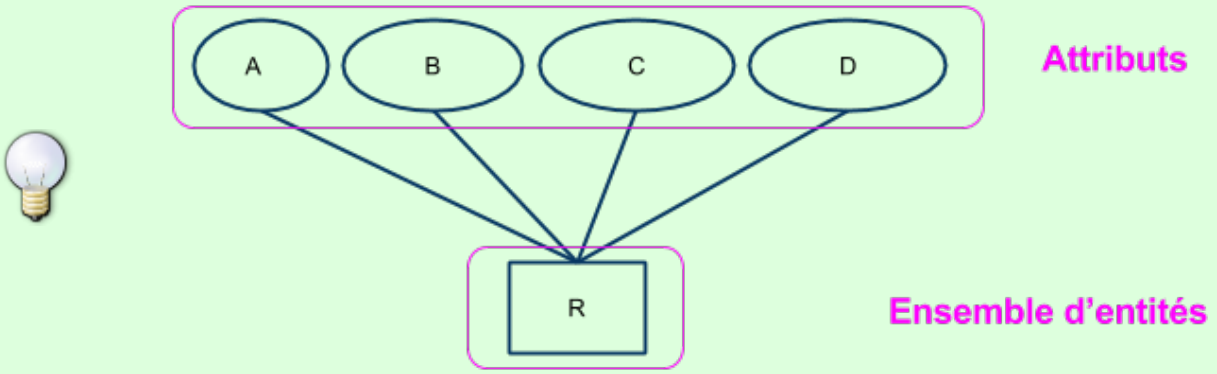
- les employés d'une firme,
- les cours de Centrale Méditerranée,
- une collection de disques,
- etc...

- Les éléments d'un ensemble d'entités sont *partiellement discernables* à travers les valeurs de leurs attributs :
  - les attributs $(A_1,...,A_m)$ servent à décrire les éléments de l'ensemble.
  - Le schéma $R$ de l'ensemble $E$ est une *application* de l'ensemble d'entités vers l'ensemble des tuples de schéma $R$

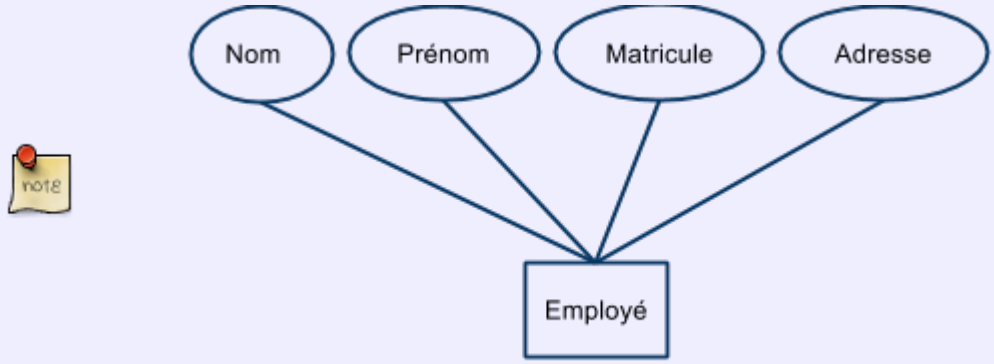
- Soit :

$$ R : \mathcal{X} \rightarrow D_1 \times ... \times D_m$$ $$ x_i \mapsto (A_1(x_i),..., A_m(x_i))$$

** représentation graphique ** :



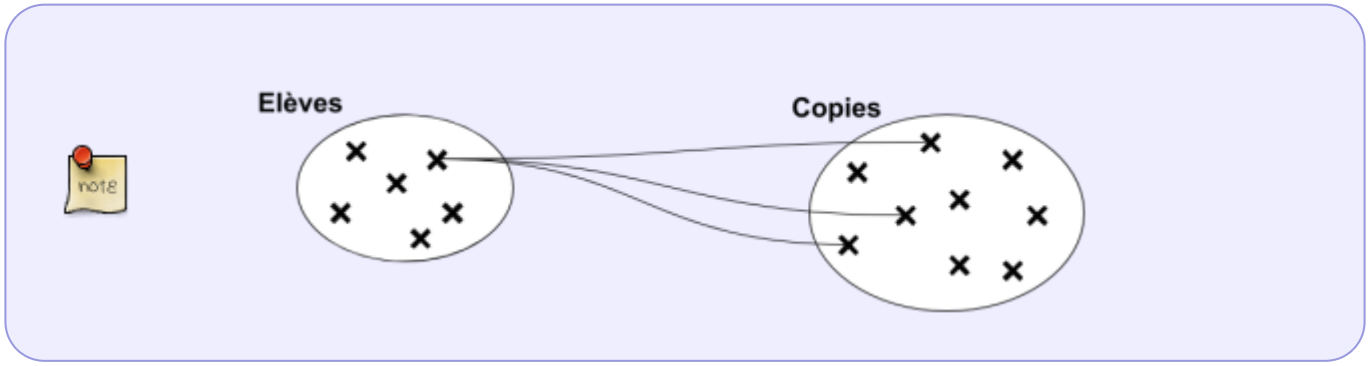

** Exemples **:

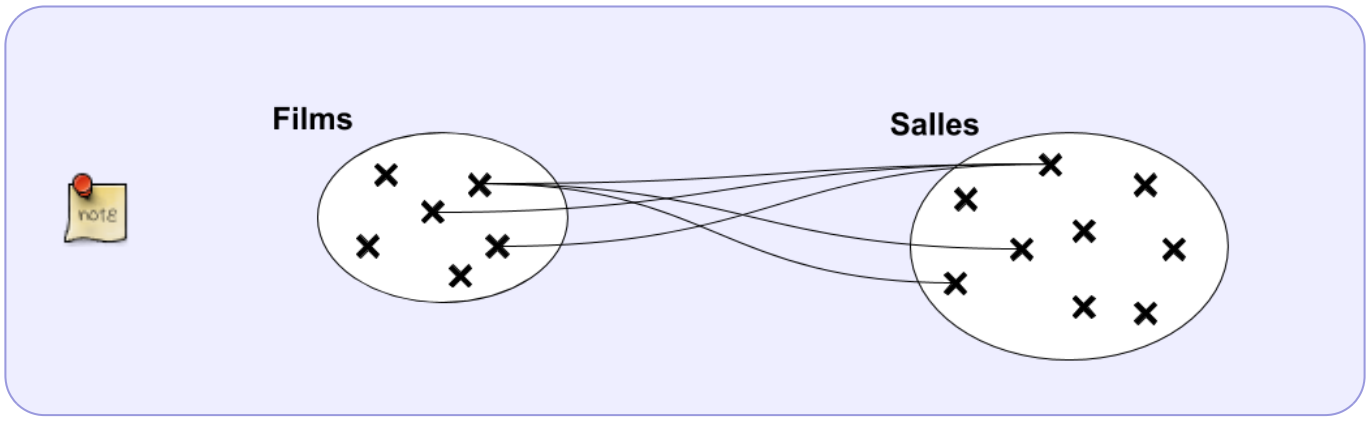


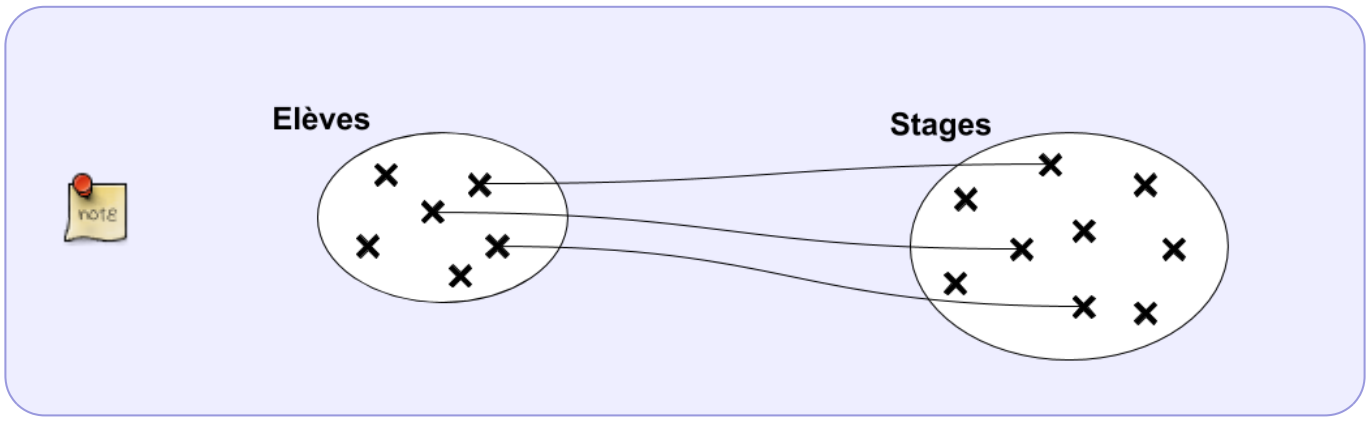

**Définitions**

Modéliser une base de données, c'est :

- Identifier les différents ensembles en interaction
- Identifier les liens de dépendance entre les différents ensembles




Les liens entre les différents ensembles sont appelés des **associations**

### Association

Une association exprime des relations de dépendance entre deux ou plusieurs ensembles d'entités.

**Définition** : Une **association** entre les ensembles $E_1$, ..., $E_k$ est un sous-ensemble du produit $E_1 \times ... \times E_k$.

Il s'agit donc d'un ensemble de k-uplets $\{..., (x_1,...,x_k), ...\}$ t.q. $x_1 \in E_1,..., x_k \in E_k$.

où $k$ est le degré de l'association :

- k=2 : association binaire
- k=3 : association ternaire
- etc...

### Rôles des associations

- ***Attribution*** : propriété, réservation, participation, supervision, auteur, rôle, pilote, ...
- ***Événements*** : achat, vente, séance, épreuve, appel, consultation, réunion, transaction, transport ...
- ***Aggrégation/Composition*** : tout/parties, contenant/contenu, supérieur/subordonné, pays/région, ...
- ***Relations entre membres*** : parenté, collaboration, cercle d'amis, ...
- ...

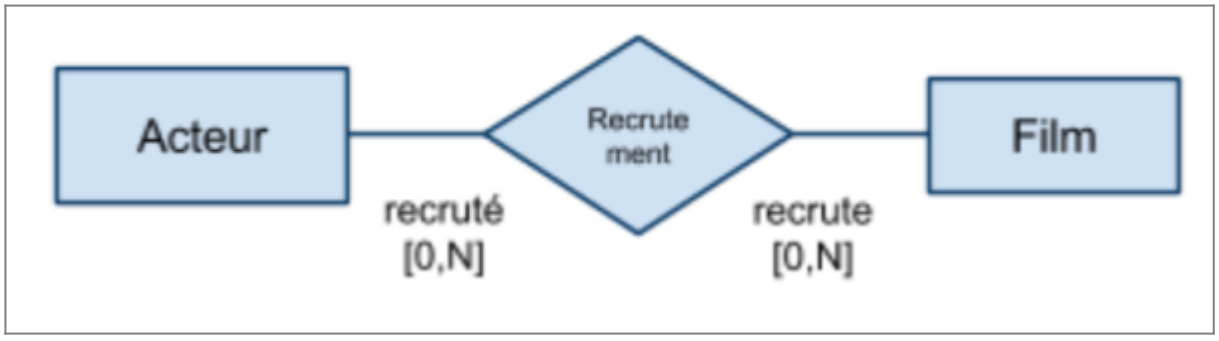
### Contraintes de cardinalité

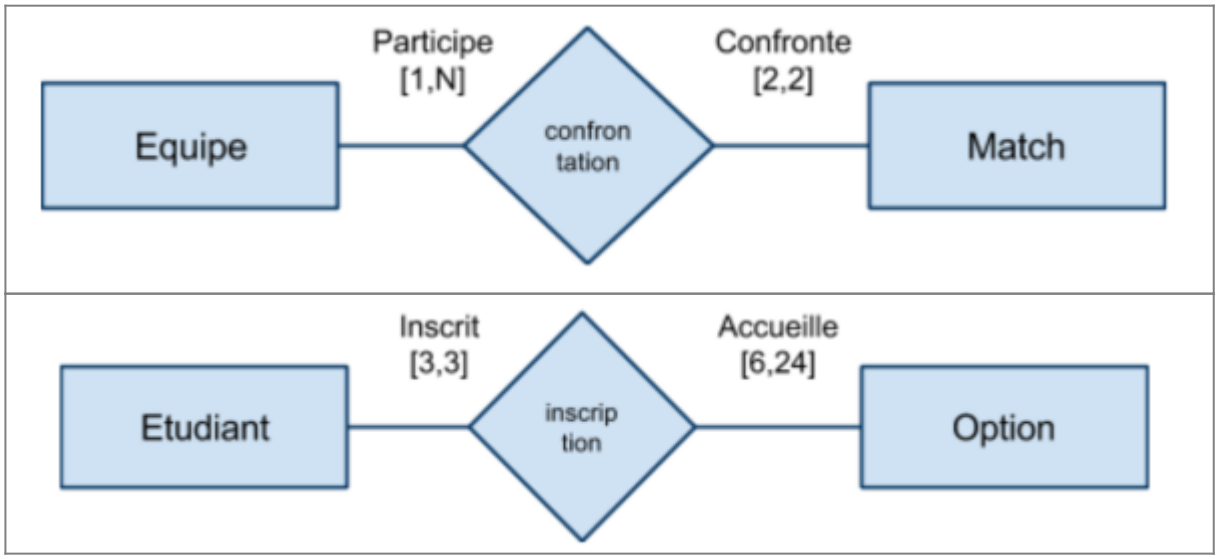
Pour chaque ensemble participant à une association, on précise dans combien d'instances de l'association chaque entité peut apparaître.

On donne en général un intervalle $[b_\text{inf},b_\text{sup}]$ qui définit le nombre d'apparitions autorisées pour chaque rôle de l'association

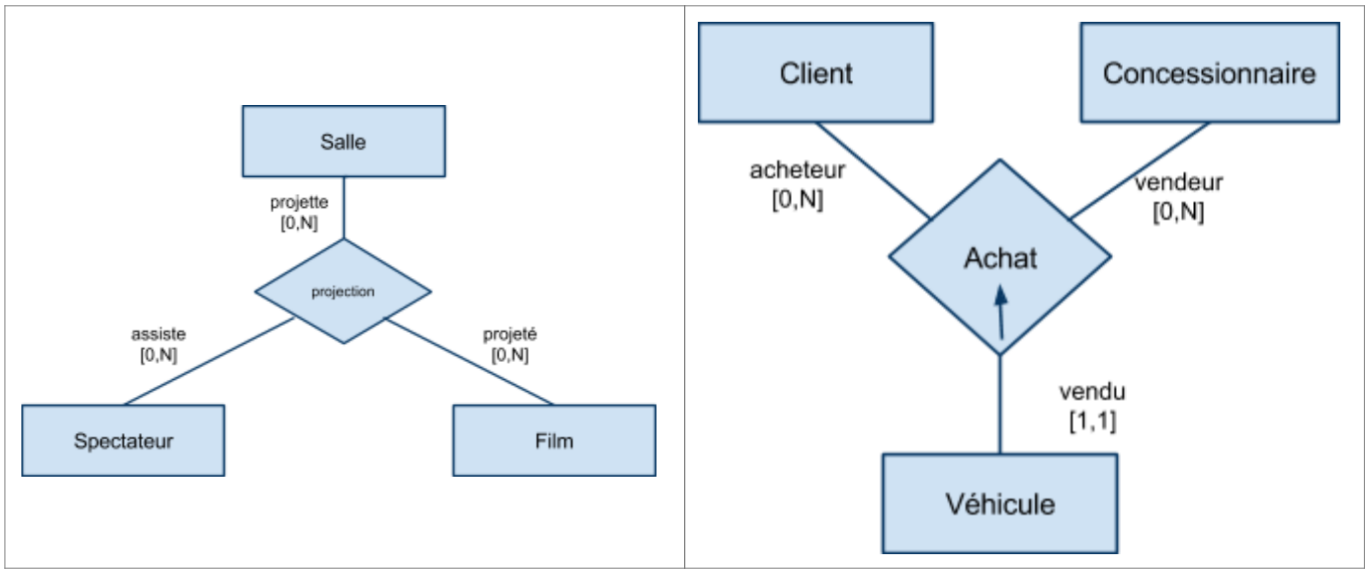
### Représentation graphique

### Associations binaires






### Associations ternaires




### Types d'associations

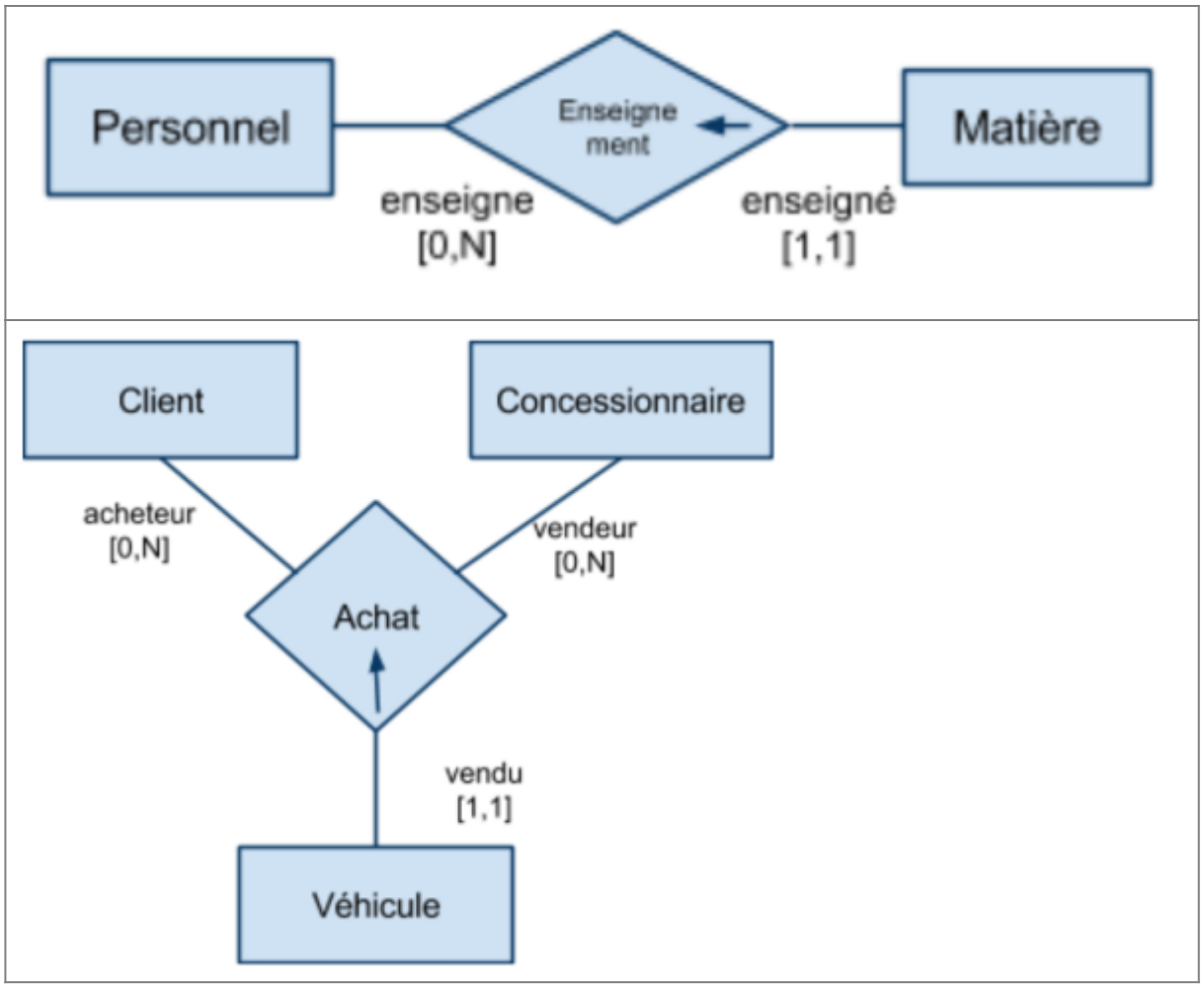
### Associations de 1 à plusieurs (fonctionnelle)

Relation non symétrique entre les deux ensembles : [...,1] d'un côté, [...,N] de l'autre. Relation de type contenant/contenu, propriétaire/objet possédé, occupant/occupé, actif/passif etc... Il s'agit du type d'association le plus "courant".



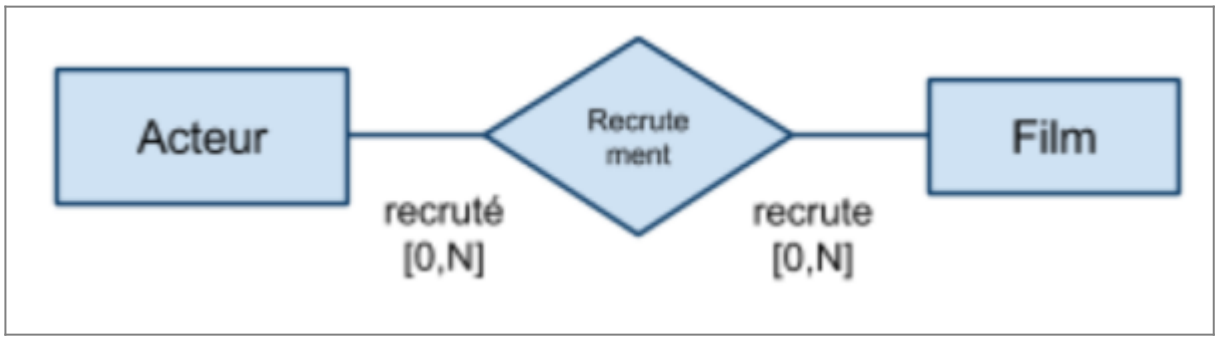
On dit parfois que l'ensemble dont la participation est unique est dit "à gauche" de l'association fonctionnelle, et celui dont la participation est multiple est "à droite", autrement dit la pointe de la flèche désigne l'ensemble de "droite":

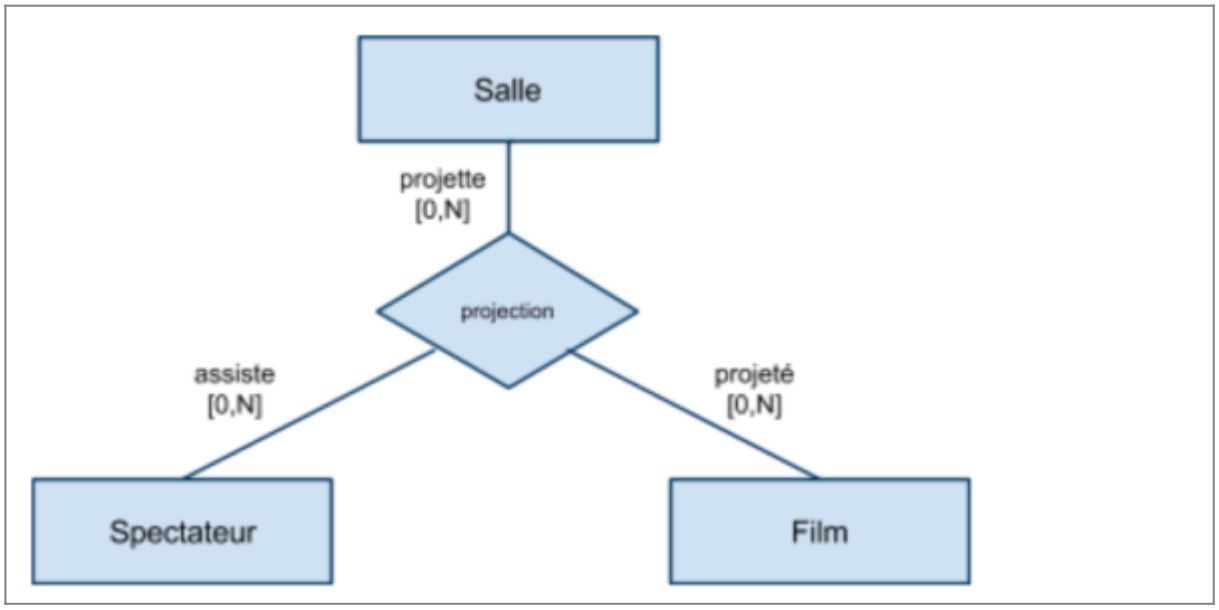
"à gauche" → "à droite"




**Associations de plusieurs à plusieurs (croisée)**

Dans une association “croisée”, les tous les lien de l’association sont de cardinalité multiple [...,N]






**Modèles Entité Associations valués**

Dans le cadre du modèle entité/association :

- les attributs des ensembles d'entités sont des *mesures*:
  - Soit $A$ un attribut de l'ensemble d'entités $\mathcal{E}$

$$ A : \mathcal{E} \rightarrow \text{dom}(A) $$

- les attributs des associations sont des *opérateurs* :
  - Soit $B$ un attribut de l'association sur $\mathcal{E} \times \mathcal{F}$

$$ B : \mathcal{E}\times \mathcal{F}\rightarrow \text{dom}(B) $$

** Mesures **

- Les mesures sont les données saisies sur les éléments d'un ensemble. Chaque mesure est associée à un attribut.
- Le schéma de l'ensemble est l'ensemble des attributs servant à caractériser ses éléments
- Les éléments de l'ensemble sont *discernables* ssi il existe un jeu de mesures différent pour chaque élément de l'ensemble
- Une *clé* est un ensemble d'attributs *minimal* (permettant de distinguer les objets) appartenant au schéma

**TODO**

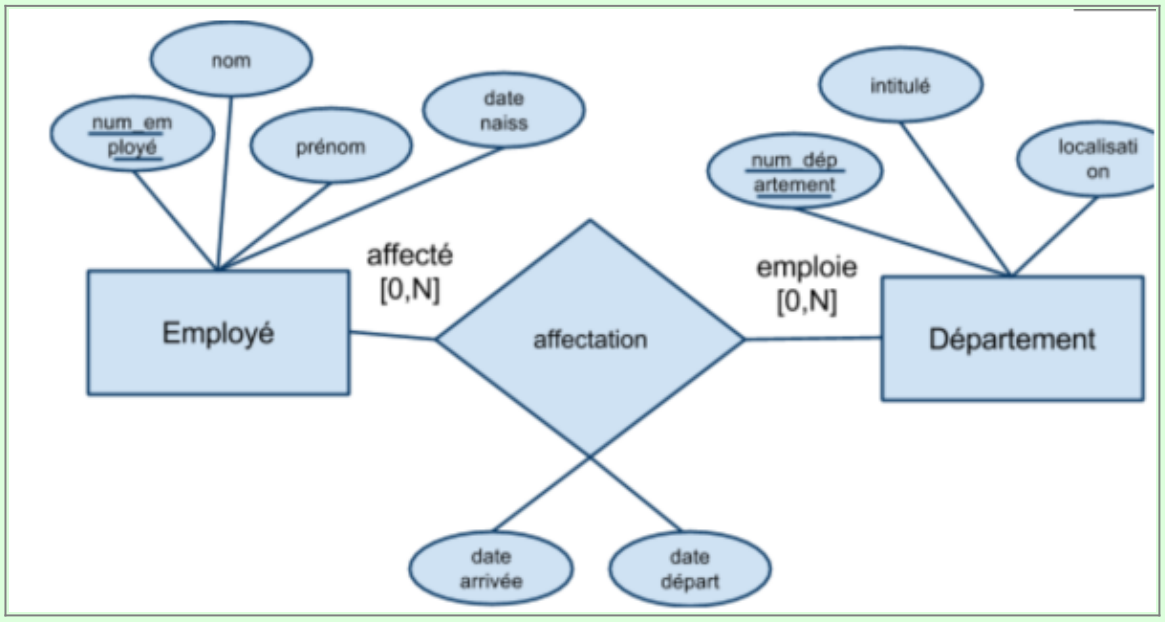
Ensembles discernables / non discernables

** Opérateurs **

- On s'intéresse ici aux associations qui représentent une "opération" (inscription, achat, embauche, affectation…).
- Lors d'une mise à jour de la base, certains événements tels que l'emprunt ou le retour d'un ouvrage, l'affectation d'un employé à un poste, ou la liste des anciens clients disparaissent.
- Il est possible de garder une trace des événements passés en mettant un (ou plusieurs) attributs sur une association.
- Ainsi, certaines associations peuvent être "datées", c'est à dire
  - avoir lieu à une date
  - ou prendre place sur une durée précise (prêt,accès temporaire, statut temporaire…)
- On peut ainsi mémoriser :
  - "Monsieur Dupont a été employé au département logistique de tant à tant."…
  - "L'étudiant X a été élève Centrale Méditerranée de telle année à telle année"…



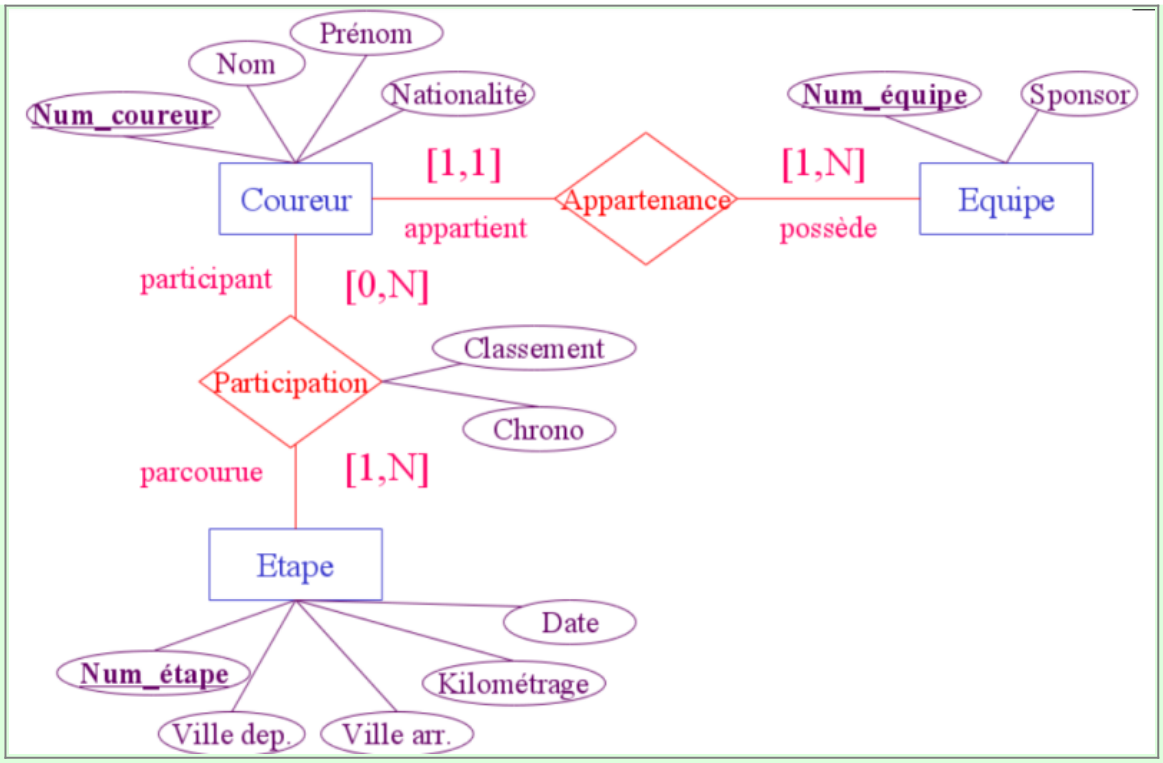
**Exemple** "Monsieur Dupont a été employé au département logistique de tant à tant."…






**Exemple**

- Chaque coureur est décrit par ses nom, prénom, nationalité et numéro de maillot.
- Chaque coureur appartient à une équipe qui possède un numéro, un sponsor associé.
- Chaque coureur participe à une ou plusieurs étapes. Une étape se caractérise par son numéro, son type (contre la montre/étape simple), ses points de départ et d'arrivée, sa date.
- A chaque étape est associée un classement d'arrivée pour chaque coureur, avec la durée totale de course.




## 2. Traduction vers le modèle relationnel

Il est possible de traduire un modèle entité/association vers un modèle relationnel (en perdant quelques propriétés).

Lors de la réalisation d'une base de données, on passe en général par les étapes suivantes:

1. Conception de la base sous forme d'un modèle entité/association.
2. Traduction sous la forme d'un modèle relationnel.
3. Normalisation (voir Normalisation d'un schéma)
4. Mise en œuvre informatique.

Un petit nombre de règles permettent de traduire un modèle entité/association vers un modèle relationnel.

- Selon ces règles, à la fois les ensembles d'entités et les associations sont transformés en schémas relationnels.
- Les liaisons et dépendances entre schémas de relation sont assurés par la définition des **clés étrangères** (attributs communs à plusieurs tables).

### Schéma de base et clé étrangère

- Un schéma (ou modèle) de bases de données est un ensemble fini de schémas de relation.
- Une base de données est un ensemble fini de relations.
- Les liens et associations entre relations entre s'expriment sous la forme de **clés étrangères**

**Définition**

- Au sein d'un schéma relationnel $R$, Une clé étrangère est un attribut (ou un groupe d'attributs) qui constitue la clé primaire d'un schéma $S$ distinct de $R$.
- La présence d'une clé étrangère au sein d'une relation $r$ de schéma $R$ introduit une contrainte d'intégrité sur les données :
  - la valeur des attributs de la clé étrangère d'un tuple de $r$ doit être trouvée dans la table s correspondante.
- On indique la présence d'une clé étrangère à l'aide de pointillés : {..., Clé étrangère, ...}

**Exemple ** <note> **Schéma de base relationnelle** :

- **Clients** ( nom_client, adresse_client, solde)
- **Commandes** ( num_Commande, nom client, composant, quantité)
- **Fournisseurs** ( nom_fournisseur, adresse_fournisseur)
- **Catalogue** ( nom_fournisseur, composant, prix )

</note>

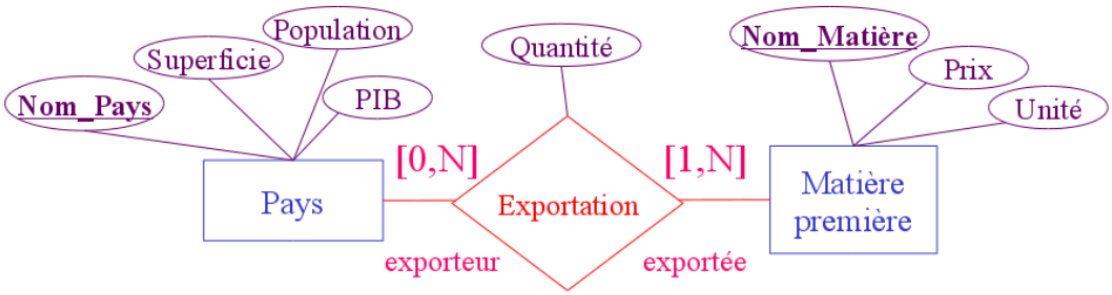
**Traduction des associations de plusieurs à plusieurs**

Une association croisée ne contient que des contraintes de cardinalité de type [..,N]. Soit $R$ une telle association et $E_1$, ..., $E_k$ les ensembles participant à l'association.

**Règle de traduction :**

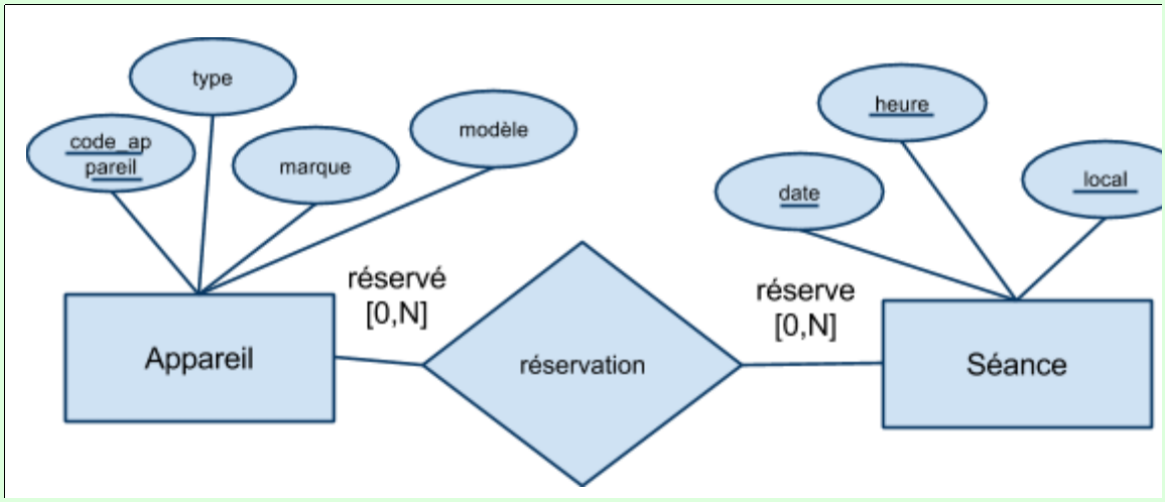
- Chaque ensemble $E_i$ est traduit par un schéma relationnel (contenant les mêmes attributs)
- L'association $R$ est traduite sous la forme d'un schéma relationnel contenant:
  - les clés primaires des ensembles participant à l'association
  - (éventuellement) les attributs propres à l'association,

Exemple :




** Traduction :**

- **Pays** (nom_pays, superficie, population, PIB )
- **Matière_première** ( nom_matière, unité, prix )
- **Exportation** (nom pays, nom matière, quantité)




**Traduction** :

- **Appareil** (code_appareil, type, marque, modèle)
- **Séance** (date, heure, local)
- **Réservation** (code appareil,date, heure, local)

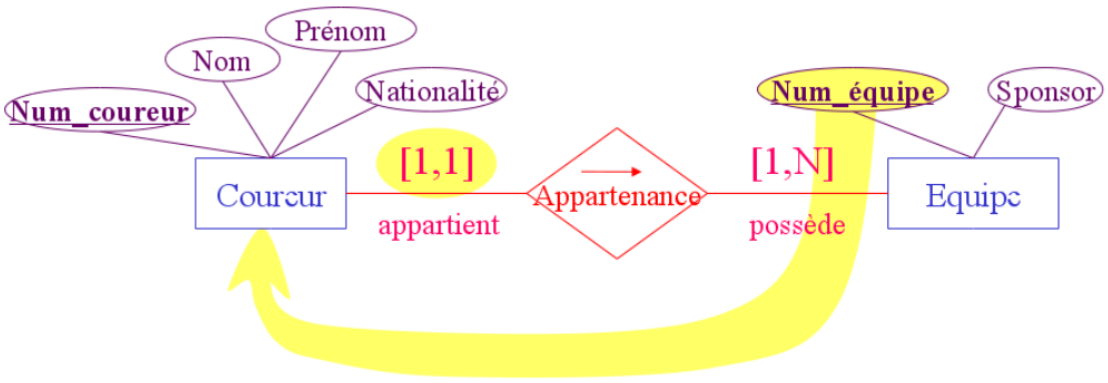
### Traduction des associations de un à plusieurs

Soit une association fonctionnelle $R$. On suppose qu'il existe au moins un ensemble $A$ de cardinalité unique [1,1] participant l'association.

**Règle de traduction**

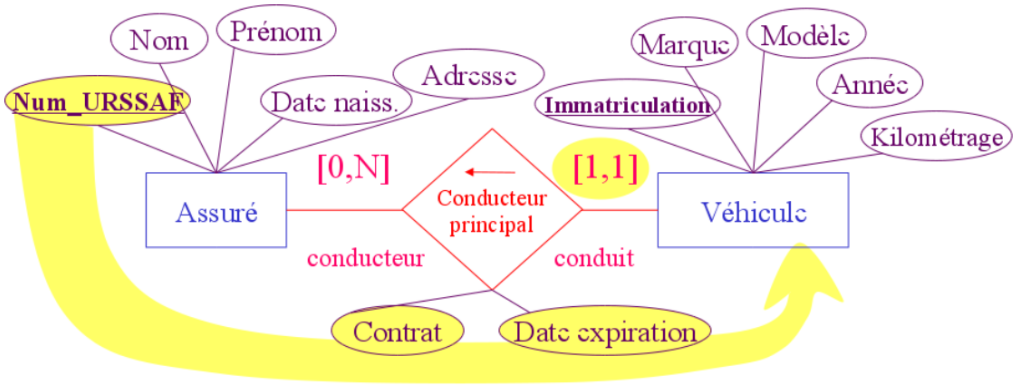
- Chaque ensemble participant est traduit sous forme de schéma relationnel
- L'association $R$ est traduite sous forme de **clé étrangère** : l'ensemble $A$ reçoit la clé primaire du (ou des) ensemble(s) dont la participation est multiple.

**Exemple** :



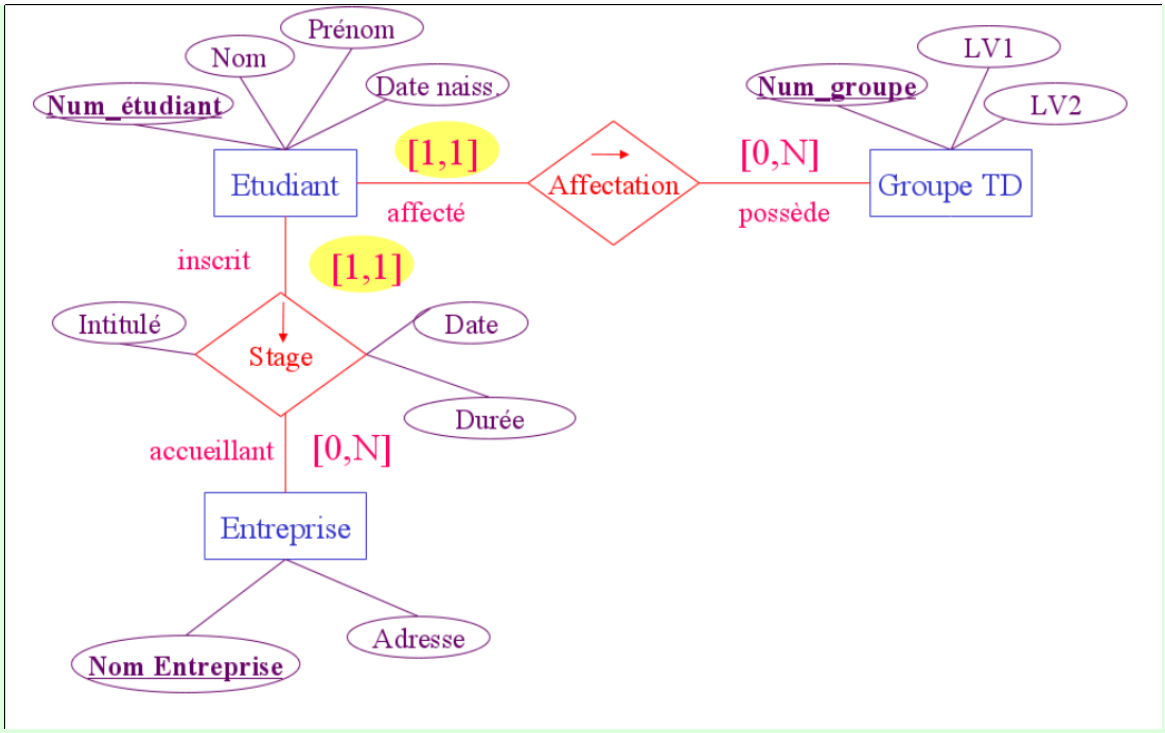

→ **Coureur** (num_coureur, nom, prénom, nationalité, num équipe)
**Equipe** (num_équipe, sponsor)

**Remarque** : lorsque l'association est valuée, les attributs de l'association sont également injectés dans la table représentant l'ensemble de gauche.




→ Assuré(num_URSSAF, nom, prénom, date_naiss, adresse)
Véhicule(immatriculation, marque, modèle, année, kilométrage, num URSSAF, contrat, date_expiration)

**Exemple**




**Traduction** :

- **Groupe_TD**( num_groupe, LV1, LV2)
- **Entreprise** ( nom_entreprise, Adresse)
- **Etudiant** ( num_etudiant, Nom, Prénom, Date_naiss, num groupe, intitulé, date, durée, nom entreprise)

### Exemple complet

**Schéma de base relationnelle** :

- **Clients** ( nom_client, adresse_client, solde)
- **Commandes** ( num_Commande, nom client, composant, quantité, montant)
- **Fournisseurs** ( nom_fournisseur, adresse_fournisseur)
- **Catalogue** ( nom_fournisseur, composant, prix )

** Réalisation ** :

**Clients** :

| **nom_client** | **adresse_client** | **solde** |
|---|---|---|
| Durand | 7, rue des Lilas | 335,00 |
| Dubois | 44, av. du Maréchal Louis | 744,00 |
| Duval | 5, place du marché | 33,00 |

**Commandes** :

| num_Commande | nom client | composant | quantité |
|---|---|---|---|
| 6674 | Dubois | micro controller | 55 |
| 6637 | Dubois | radio tuner | 2 |
| 6524 | Durand | transistor | 4 |
| 6443 | Duval | micro controller | 7 |

**Fournisseurs** :

| nom_fournisseur | adresse_fournisseur |
|---|---|
| Sage | 33, College street, London |
| MoxCom | 77 Ashley square,Mumbay |

**Catalogue** :

| nom_fournisseur | composant | prix |
|---|---|---|
| Sage | transistor | 4,4 |
| MoxCom | micro controller | 3,7 |
| MoxCom | radio tuner | 7,0 |

# Recherche d'information

### Rappel : modèle relationnel

Les données sont structurées en **tables**, contenant des **tuples** organisés comme séquence de **valeurs**, i.e. :

- Une **base de données** est un ensemble de **tables** (parfois appelées relations).
- Une **table** est un ensemble de **tuples** (parfois appelés enregistrements).
- Un **tuple** est un ensemble de **valeurs** (parfois appelés « champs » ou attributs).

### Exemple complet

**Schéma de base relationnelle** :

- **Clients** ( nom_client, adresse_client, solde)
- **Commandes** ( num_Commande, nom client, composant, quantité, montant)
- **Fournisseurs** ( nom_fournisseur, adresse_fournisseur)
- **Catalogue** ( nom_fournisseur, composant, prix )

** Réalisation ** :

**Clients** :

| **nom_client** | **adresse_client** | **solde** |
|---|---|---|
| Durand | 7, rue des Lilas | 335,00 |
| Dubois | 44, av. du Maréchal Louis | 744,00 |
| Duval | 5, place du marché | 33,00 |

**Commandes** :

| **num_Commande** | **nom client** | **composant** | **quantité** |
|---|---|---|---|
| 6674 | Dubois | micro controller | 55 |
| 6637 | Dubois | radio tuner | 2 |
| 6524 | Durand | transistor | 4 |
| 6443 | Duval | micro controller | 7 |



**Fournisseurs** :

| **nom_fournisseur** | **adresse_fournisseur** |
|---|---|
| Sage | 33, College street, London |
| MoxCom | 77 Ashley square,Mumbay |

**Catalogue** :

| **nom_fournisseur** | **composant** | **prix** |
|---|---|---|
| Sage | transistor | 4,4 |
| MoxCom | micro controller | 3,7 |
| MoxCom | radio tuner | 7,0 |

## 1. Gestionnaire de Bases de données

Un SGBD (Système de Gestion de Bases de Données) est un programme qui gère une (ou des) base(s) de données. Il s'agit d'un programme optimisé afin d'accélérer l'accès et rationaliser le traitement d'un grand ensemble d'enregistrements stockés sur un support informatique.

Le fonctionnement d'un tel programme repose sur :

- des méthodes de conception ensemblistes fondées sur le modèle relationnel (description des ensembles d'enregistrements et des relations entre ces ensembles)
- un langage de requête permettant de consulter et mettre à jour les données stockées dans la base
- des algorithmes de stockage et de classement efficace (afin d'accélérer les temps de recherche)

Il existe enfin un administrateur de bases de données. Celui-ci doit :

- installer la base de données,
- gérer sa sauvegarde régulière,
- garantir sa sécurité et

- gérer les droits des utilisateurs de la base.

### SQL (Structured Query Language)

**SQL** :

- est un langage de création et d'interrogation de bases de données,
- supporté par la plupart des systèmes de gestion de bases de données relationnelles du marché.



**Structured query language** (SQL), ou langage structuré de requêtes, est un langage informatique standard et normalisé, destiné à interroger ou manipuler une base de données relationnelle. Les instructions SQL se répartissent en trois familles distinctes :

- un langage de définition de données (LDD, ou en anglais DDL, Data definition language), qui permet la description de la structure de la base (tables, vues, index, attributs, ...).
- un langage de manipulation de données (LMD, ou en anglais DML, Data manipulation language), c'est la partie la plus courante et la plus visible de SQL, qui permet la manipulation des tables et des vues.
- et un langage de contrôle de données (LCD, ou en anglais DCL, Data control language), qui contient les primitives de gestion des transactions et des privilèges d'accès aux données.



Exemple de définition de schéma de table avec clé étrangère en SQL :

```
  CREATE TABLE Commande (
    num_commande INTEGER NOT NULL,
    nom_client VARCHAR(30),
    nom_fournisseur VARCHAR(30),
    composant VARCHAR(30),
    quantité INTEGER,
    montant DECIMAL(12,2) NOT NULL,
    PRIMARY KEY (num_commande),
    FOREIGN KEY (nom_client) REFERENCES Client,
      FOREIGN KEY (nom_fournisseur, composant) REFERENCES
Catalogue);
```



**Lecture/écriture en SQL**

- Création :

```
INSERT INTO Client VALUES ("Durand", "7, rue des Lilas" ,
335.00);
```

- Mise à jour

```
UPDATE Client SET adresse = "9, rue des Lilas" WHERE nom_client="Durand";
```



- Suppression

```
DELETE FROM Client WHERE nom_client="Durand";
```

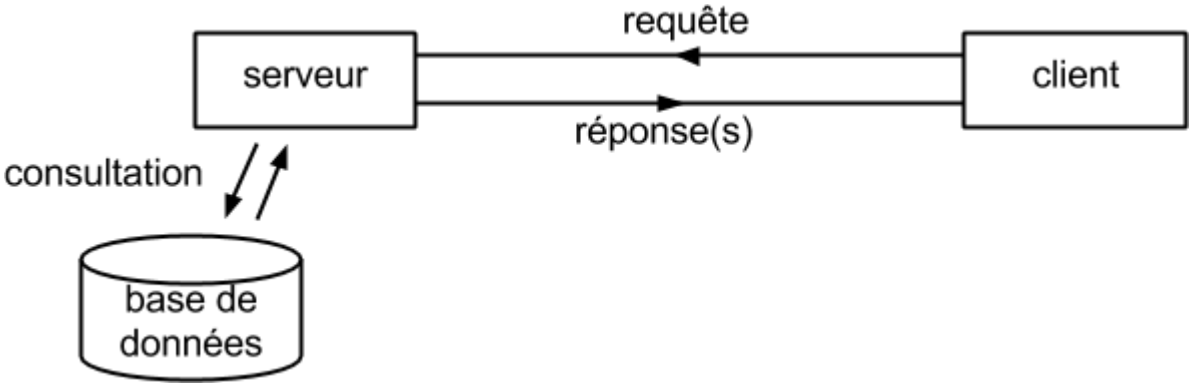
Voir : sql.pdf

## 2. Interrogation des Bases de Données

Interroger une base de données , c'est sélectionner certaines données parmi l'ensemble des données proposés.

En informatique, une requête (en anglais query) est une demande de consultation ou d'édition, effectuée par un programme *client* à l'attention d'un programme *serveur*.

- Le programme **client** représente l'utilisateur, il s'agit du programme qui enregistre la demande de l'utilisateur, la transmet au serveur, puis met en forme visuellement la réponse du serveur.
- Les données sont centralisées au niveau du **serveur**, chargé de la gestion, de la manipulation et du stockage des données.Il traite la requête, consulte les données et transmet le résultat au client.




### Consultation des données

La requête peut être une simple référence vers un fichier, ou être l'expression d'une recherche plus spécifique (consultation de certaines fiches d'un fichier, croisement d'information (entre plusieurs fichiers), etc...). Dans ce cas, il est nécessaire d'utiliser un langage de requête (le plus souvent SQL).

On distingue quatre grands types de requêtes (approche "CRUD"):

- **Création** (***C**reate*) : ajout de nouvelles données dans la base
- **Lecture/recherche** (***R**ead*) : consulation du contenu de la base
- **Mise à jour** (***U**pdate*) : changement du contenu existant
- **Suppression** (***D**elete*) : suppression des données obsolètes

Lors d'une consultation de type lecture/recherche, il y a souvent plusieurs réponses qui correspondent à la demande. Le résultat d'une requête prend donc la forme d'un ensemble de réponses. Ces réponses sont éventuellement classées, selon la valeur d'un certain identifiant, ou selon le degré de pertinence.



Exemples :

- requêtes http : demande de consultation d'une page web ( = référence vers un fichier)
- moteur de recherche : recherche de pages contenant les mots-clés spécifiés
- bases de données : utilisation d'un langage de requête :

```
SELECT *
FROM  Eleves
WHERE NOM = 'Dugenou'
```

Exemples :

- **croisement de facteurs de recherche :**
  - Donner la liste des pays exportateurs de pétrole
  - Liste des musiciens jouant à la fois du piano et du violon
  - Liste des artistes français disque d'or en 1977
  - Liste des suspects châtain taille moyenne présents à Poitiers la nuit du 12 au 13 février
- ** Analyse des données **
  - faire ressortir des corrélations (exemple : type d'habitat/intentions de vote),
  - pour des enquêtes de consommation, du marketing ciblé ...
- **Information personnalisée :**
  - ne retenir que les informations utiles à un instant et pour une personne donnée.
  - Emploi du temps de l'élève X pour la semaine Y.
  - Factures impayées du client Z



Lors d'une consultation de type lecture/recherche,

- il y a souvent plusieurs réponses qui correspondent à la demande.
- Le résultat d'une requête prend donc la forme d'un ensemble de réponses.
- Ces réponses sont éventuellement classées,
  - selon la valeur d'un certain identifiant,
  - ou selon le degré de pertinence.



L'**algèbre relationnelle**

- propose un certain nombre d'opérations ensemblistes :
  - Intersection,
  - Union,
  - Projection,
  - Différence,
  - ...

- qui, à partir d'un ensemble de relations, permettent de construire de nouvelles relations.
- La relation nouvellement construite contient le résultat de la **requête**

## 3. Opérateurs mono-table

Extraction d'information à partir d'une table unique :

- projection $\pi$ = extraction de colonnes
- sélection $\sigma$ = extraction de lignes

### 3.1 Projection : π

**Projection**

- Soit $r$ une relation de schéma $R$.
- Soit $S$ un ensemble d'attributs, avec $S$ ⊆ $R$

La **projection** $\pi_S(r)$ est une nouvelle relation de schéma $S$ obtenue à partir des éléments de $r$ restreints au schéma $S$ $$\pi_S(r) = \{t(S)|t \in r\}$$

(avec $t(S)$ la restriction de $t$ au schéma $S$)

**Exemple Catalogue** :

| nom_fournisseur | adresse_fournisseur | composant | prix |
|---|---|---|---|
| Sage | 33, College street, London | transistor | 4,4 |
| MoxCom | 77 Ashley square,Mumbay | micro controller | 3,7 |
| MoxCom | 77 Ashley square,Mumbay | radio tuner | 7,0 |

Requete : *Donner la liste des fournisseurs (avec leur adresse)*: $$u = \pi_\text{nom_fournisseur, adresse_fournisseur} (\text{Catalogue})$$

$\rightarrow$ **u** :

| nom_fournisseur | adresse_fournisseur |
|---|---|
| Sage | 33, College street, London |
| MoxCom | 77 Ashley square,Mumbay |

### 3.2 Sélection : σ

**Condition sur R**

- On considère le schéma $R (A_1, ..., A_n)$
- Une condition $F$ sur $R$ :
  - est un ensemble de contraintes sur les valeurs des attributs $A_1$, ..., $A_n$
  - construites à l'aide d'opérateurs booléens classiques :
    - ∧(et),
    - ∨(ou),
    - ¬(non),
    - =, ≠, >,<, ≥ ,≤, ...
    - et de valeurs numériques ou de texte.

**Exemples** : $$ F = (A_1 = 3) ∧ (A_1 > A_2) ∧ (A_3 ≠ 4)$$ $$ F = (A_1 = 2) ∨ (A_2 = "Dupont")$$

**Sélection**

- Soit $r$ une relation de schéma $R$
- Soit $F$ une condition sur $R$

La **sélection** $σ_F(r)$ est une nouvelle relation de schéma $R$ , constituée de l'ensemble des enregistrements de $r$ qui satisfont la condition $F$.

$$σ_F(r) = \{ t ∈ r | F( t ) \text{est vrai} \}$$

**Exemple** :

Requête : *Donner la liste des fournisseurs qui vendent des micro-controleurs*

$$u = Π_\text{nom_fournisseur}( σ_\text{Composant = micro controller} ( \text{Fournisseur} ))$$ **u** :

| **nom_f** |
|---|
| Moxcom |

**Exemple**

**Pays** :

| **nom_pays** | **superficie** | **population** | **PIB/hab** |
|---|---|---|---|
| Algérie | 2.300.000 | 31.300.000 | 1630$ |
| Niger | 1.200.000 | 11.400.000 | 890$ |
| Arabie Saoudite | 2.150.000 | 24.300.000 | 8110$ |

Requête : *Donner la liste des pays dont le PIB/hab est > 1000$* $$u = Π_\text{nom_pays}( σ_\text{PIB/hab > 1000 } ( \text{Pays} ))$$

**u** :



| nom_pays |
| --- |
| Algérie |
| Arabie Saoudite |

### 3.3 Structure d'une requête SQL

```
SELECT    A1,A2, …, An    // liste d'attributs
FROM      R               // nom de la TABLE
WHERE     F               // condition sur les attributs
```

cette requête est semblable à :

- une sélection algébrique $σ_F$
- suivie par une projection algébrique $Π_{A1, ..., An}$

soit : $$Π_{A1, ..., An}( σ_F ( R ))$$

**Exemples :**

- *Qui fournit des transistors ?*

```
SELECT nom_fournisseur
FROM Fournisseur
WHERE composant = 'transistor';
```

- *Liste de toutes les commandes de transistors :*

```
SELECT *
FROM Commandes
WHERE composant = 'transistor'
```

- *Qui fournit des micro-controleurs à moins de 5$?*

```
SELECT nom_fournisseur
FROM Catalogue
WHERE composant = 'micro controller' AND prix < 5
```

### 3.4 Aspects algorithmiques

**Algo de sélection :**

```
pour tout tuple t de r:
  si t obéit à la condition  F:
        ajouter t à la réponse
```

complexité : O(n)



- On parle de recherche ciblée lorsque le nombre d'éléments obéissant à la condition F est a priori très faible :
  - exemple : on cherche le salaire de Mr "Dupont".
- On parle de recherche extensive lorsque l'ensemble de valeurs obéissant à la condition F est a priori élevé.
  - exemple : les employés dont le salaire est < 2000 euros

**Rappel** :

Organisation des données sous forme de tableaux bidimensionnels :

** Schémas de données **

- Les données sont organisées sous forme de **tuple**
- A un tuple correspond en général un **schéma de données**, qui permet d'interpréter les valeurs présentes dans le tuple.

| SCHEMA : | Nom | Prénom | Adresse | Âge |
|---|---|---|---|---|
| DONNEES : | Dubois | Martine | 29, rue du Verger, Orléans | 22 |

** Relation **

Une *relation* est un ensemble de tuples, chaque tuple obéissant à un même schéma $R$. La relation est la description logique d'un *tableau de données*.

**Tableau de données**

| Nom | Prénom | Adresse | Âge | |
|---|---|---|---|---|
| | | | | schéma |
| Dubois | Martine | 29, rue du Verger, Orléans | 22 | |
| Gilbert | Jonas | 8, rue des Fleurs, Blois | 23 | |
| Dalban | Pierre | 13, av. du Général, Privas | 22 | tuple |
| ... | ... | ... | ... | |
| Manoukian | Marianne | 55, place des Bleuets, Aubagne | 24 | |

Remarque : dans le cas d'une recherche ciblée, la présence d'un index sur le critère de recherche permet d'accélérer la sélection :

```
pour tout a ∈ F:
  t ← Index(a)
  ajouter t à la réponse
```

## 4. Opérateurs multi-tables

Principe : recoupement d'informations présentes dans plusieurs tables :

- Croisement des critères de sélection : **Jointure**
- Recherche ciblée : **Division**

### 4.1 La jointure : ⋈



**Union de deux éléments :**

- Soient les relations r et s de schémas R et S.
- On note R ⋂ S la liste des attributs communs aux deux schémas et R ⋃ S la liste des attributs appartenant à R ou à S.
- soit t ∈ r et q ∈ s tels que t(R ⋂ S) = q(R ⋂ S)

On note t ⋃ q le tuple formé des valeurs de t et de q étendues au schéma R ⋃ S



**Produit cartésien **

- Soient r et s (de schémas R et S), avec R ⋂ S = Ø

Le produit cartésien r × s est une nouvelle table de schéma R ⋃ S combinant les tuples de r et de s de toutes les façons possibles : $$r × s = \{t \cup q : t \in r, q \in s\}$$

- La **jointure** est une opération qui consiste à effectuer un produit cartésien des tuples de deux relations pour lesquelles certaines valeurs correspondent.
- Le résultat de l'opération est une nouvelle relation.



**Jointure**

- Soient r et s (de schémas R et S), avec R ⋂ S ≠ Ø
- La **jointure** r ⋈ s est une nouvelle table de schéma R ⋃ S combinant les tuples de r et de s ayant des valeurs communes pour les attributs communs.

$$r ⋈ s = \{t \cup q : t∈ r, q∈ s, t(R \cap S) = q(R \cap S)\}$$

**Exemple**

**Matière_première :**

| nom_matière | unité | prix |
|---|---|---|
| pétrole | baril | 45$ |
| gaz | GJ | 3$ |
| uranium | lb | 12$ |

**Exportations :**



| nom_pays | nom_matière | quantité |
|---|---|---|
| Algérie | pétrole | 180.000 |
| Algérie | gaz | 20.000 |
| Niger | uranium | 30.000 |
| Arabie Saoudite | pétrole | 2.000.000 |
| Arabie Saoudite | gaz | 750.000 |

**Matière_première ⋈ Exportations **:

| nom_pays | nom_matière | quantité | unité | prix |
|---|---|---|---|---|
| Algérie | pétrole | 180.000 | baril | 45$ |
| Algérie | gaz | 20.000 | GJ | 3$ |
| Niger | uranium | 30.000 | lb | 12$ |
| Arabie Saoudite | pétrole | 2.000.000 | baril | 45$ |
| Arabie Saoudite | gaz | 750.000 | GJ | 3$ |

**Exemples de requêtes **

- "*Donner la liste des PIB/hab des pays exportateurs de pétrole*" :

$$Π_\text{PIB/hab}( σ_\text{nom_matière = pétrole} ( \text{Pays} ⋈ \text{Exportations} ))$$

**Schéma de base relationnelle** :



- **Clients** ( nom_client, adresse_client, solde)
- **Commandes** ( num_Commande, nom client, nom fournisseur, composant, quantité, montant)
- **Fournisseurs** ( nom_fournisseur, adresse_fournisseur)
- **Catalogue** ( nom_fournisseur, composant, prix )

- "*Donner le nom et l'adresse des clients qui ont commandé des micro controleurs*" :

$$Π_\text{nom_client,adresse_client}( σ_\text{composant = 'micro-controller'} ( \text{Client} ⋈ \text{Commandes} ))$$

### 4.2 Requêtes multi-tables en SQL

```
SELECT    A1,A2, …, An       // liste d'attributs
FROM      R1, …, Rm          // liste de TABLES
WHERE     F1 AND … AND Fl    // liste de conditions sur les attributs
                             // (en particulier conditions sur les attributs
                             // sur lesquel  s'effectue la jointure)
```

Pour exprimer la jointure sur l'attribut 'Aj' commun aux tables 'R1' et 'R2', on écrira : 'R1.Aj = R2.Aj'

**Exemples :**

- **Approche prédicative : **

```
SELECT PIB_par_hab
FROM Pays NATURAL JOIN Exportations
WHERE nom_matiere = 'petrole'
```

```
SELECT PIB_par_hab
FROM Pays, Exportations
WHERE nom_matiere = 'petrole'
AND Pays.nom_pays = Exportations.nom_pays
```

- **Approche ensembliste : **

```
SELECT PIB_par_hab
FROM Pays
WHERE nom_pays IN (
  SELECT  nom_pays
  FROM Exportations
  WHERE nom_matiere = 'petrole'
  )
```

### 4.3 Aspects algorithmiques et optimisation

Lors d'une opération de jointure, on distingue en général la "table de gauche" de la "table de droite".



Dans le modèle Entité/Association,

- La table de gauche est celle qui est de cardinalité unique,
- et la table de droite est celle qui est de cardinalité multiple (définissant une relation fonctionnelle gauche → droite)
- L'attribut servant pour la jointure est donc clé étrangère de la table de gauche et clé primaire de la table de droite.

Il y a deux stratégies possibles pour faire une jointure :

- **double boucle** : c'est l'algorithme le plus simple. La table de droite est examinée une fois pour chaque tuple de la table de gauche. Cette stratégie est facile à implémenter mais peut être très coûteuse. Toutefois, si la table de droite est indexée sur l'attribut qui sert pour la jointure (cas le plus courant), elle peut donner de bons résultats;
- **tri-fusion** : chaque table est triée sur les attributs de jointure avant que la jointure soit effectuée. Une fois le tri fait, les tables peuvent être fusionnée : chaque table n'est examinée qu'une seule fois (ici, la jointure n'est pas coûteuse, c'est bien sûr sur l'opération de tri que tout est reporté).

** Exemple :**

On considère les schémas R(A1,A2,A3) et S(A3,A4,A5) ayant l'attribut A3 en commun :

- L'attribut A3 est une clé étrangère du schéma R.
- R est "à gauche" (contient la clé étrangère), et S est "à droite".

Soient r et s deux tables obéissant aux schémas R et S, et de taille |r| et |s|.

*Algo naif :*

```
Pour chaque tuple t de r:
    Pour chaque tuple q de s:
       Si t(A3)=q(A3):
          ajouter t U q à la réponse
```

Complexité : O (|r| ×|s|)

Si la table de droite est indexée sur l'attribut A3 qui sert pour la jointure, l'algo suivant peut donner de bons résultats :

```
Pour chaque tuple t de r: # table de gauche
  a ← t(A3)
  u ← Index(a) # index sur la table de droite
  ajouter t U u à la réponse
```

- si la recherche dans l'index est en est en log(|s|)
- Complexité : O (|r| × log |s|)

### 4.4 La division



**Division**

- Soient r (de schémas R) et s (de schémas S), avec S ⊆ R :

La division r ÷ s est la relation (table) u de schéma R-S maximale contenant des tuples tels que u × s ⊆ r (avec × représentant le produit cartésien)

$$r ÷ s = \{ t | ∀ q ∈ s, t \cup q ∈ r\}$$

→ on cherche les éléments de t qui "correspondent" à s

Exemples :

| Nom_Pays | Nom_matière |
|---|---|
| Algérie | Pétrole |
| Algérie | Gaz |
| Niger | Uranium |
| Arabie Saoudite | Pétrole |
| Arabie Saoudite | Gaz |

÷

| Nom_matière |
|---|
| Pétrole |
| Gaz |

=

| Nom_Pays |
|---|
| Algérie |
| Arabie Saoudite |

Exemples :

- Liste des pays qui exportent *tous* les types de matières premières
- Liste des pays qui exportent les *mêmes* matières premières que l'Arabie Saoudite
- Liste des élèves présents à tous les cours magistraux d'informatique de première année

## 5. Recherches composées

- Certaines requêtes, peuvent être le résultat de la combinaison de plusieurs critères de recherche
- La combinaison de résultats est généralement réalisée à l'aide des opérations ensemblistes classiques (intersection, union...) pour exprimer «et», «ou», «non»...
- Pour alléger les formules, il est possible d'utiliser des tables intermédiaires.



**Union**

- Soient r1 et r2 deux tables de schéma R.

L'**union** r1 U r2 est une nouvelle table de schéma R constituée de l'ensemble des enregistrements qui appartiennent à r1 ou à r2: $$r1 \cup r2 = { t ∈ r1} \cup { t ∈ r2}$$



**Intersection**

- Soient r1 et r2 deux tables de schéma R.

L'**intersection** r1 ⋂ r2 est une nouvelle table de schéma R constituée de l'ensemble des enregistrements qui appartiennent à r1 et à r2: $$r1 \cap r2 = \{ t ∈ r1\} \cap \{ t ∈ r2\}$$



**Différence ** * Soient r1 et r2 deux tables de schéma R. La **différence** r1 - r2 est une nouvelle table de schéma R constituée de l'ensemble des enregistrements qui appartiennent à r1 mais pas à r2: $$r1 - r2 = \{ t ∈ r1\} - \{ t ∈ r2\}$$

**Exemples :**

- Donner la liste des pays qui exportent à la fois du gaz et du pétrole :

$$\pi _{Pays} σ_\text{matière = gaz} (\text{Exportations}) \cap \pi _{Pays} σ_\text{matière = pétrole} (\text{Exportations})$$ en SQL :

```
SELECT pays FROM Exportations
WHERE matière = 'gaz'
INTERSECT (     SELECT pays FROM EXPORTATIONS
                WHERE matière = 'pétrole');
```

- Donner la liste des pays qui exportent du gaz mais pas du pétrole :

$$\pi _{Pays} σ_\text{matière = gaz} (\text{Exportations}) - \pi _{Pays} σ_\text{matière = pétrole} (\text{Exportations})$$ en SQL :

```
SELECT pays FROM Exportations
WHERE matière = 'gaz'
EXCEPT (     SELECT pays FROM EXPORTATIONS
            WHERE matière = 'pétrole');
```

* Donner la liste des clients qui commandent uniquement des produits 'Moxcom' :
$$\pi_{nom_client}Client - \pi_{nom_client} \sigma_{fournisseur \neq 'Moxcom'} Client ⋈ Commande$$ en SQL :

```
SELECT nom_client FROM Client
EXCEPT (    SELECT client FROM Client NATURAL JOIN Commande
            WHERE fournisseur <> 'Moxcom');
```

## 6. Agrégation

### Généralités

L'analyse des données a pour but de recomposer l'information contenue dans les données recueillies afin d'en fournir une vue plus synthétique, ou d'extraire des informations qui n'apparaissent pas de façon explicite dans la série de mesures initiale.



**Rappel**

On distingue classiquement deux grandes catégories de données :

- données **quantitatives**:
  - numérique entier ou réel, discrètes ou continues, bornées ou non. ex: poids, taille, âge, taux d'alcoolémie,...
  - temporel : date, heure
  - numéraire
  - etc...
- données **qualitatives**:
  - de type vrai/faux (données booléennes). ex: marié/non marié, majeur/non majeur



- de type appartenance à une classe. ex: célibataire/marié/divorcé/veuf, salarié/chômeur/retraité etc...
- de type texte (autrement dit “chaîne de caractères”). ex: nom, prénom, ville,...

Les données qualitatives:

- définissent l'appartenance à une catégorie
- permettent de définir des **classes** au sein d'un ensemble de données

### Partition

L'agrégation consiste à partitionner les données en classes selon la **valeur** d'un attribut.



**Rappel**: partition d'un ensemble $E$

Une partition d'un ensemble $E$ est un ensemble de parties de $E$ tel que:

- aucune de ces parties n'est vide
- quel que soit le choix de deux de ces parties, elles sont disjointes
- la réunion de toutes ces parties est l'ensemble $E$



Il est supposé que le partitionnement s'effectue sur des attributs pour lequel le nombre de valeurs possibles est fini.

Si A est l'attribut considéré:

- A est un attribut "qualitatif"
- Le domaine de valeurs de A $\text{dom}(A)$ est fini
- $\text{dom}(A)$ définit un index de partitionnement

### Index de partitionnement

Un index de partitionnement:

- est constitué d'un ensemble fini de classes $\mathcal{C} = \text{dom}(A)$
- à chaque classe $c \in \mathcal{C}$ sont associées plusieurs références de la table de données $r$
- autrement dit :

$$ \forall \text{c} \in \mathcal{C}, c \rightarrow r_c = \{t_1, t_2, ...\} \subset r$$

- On dit que les données sont organisées en *classes* : chaque tuple appartient à une classe unique $c$

- L'ensemble de classes $\mathcal{C}$ définit une partition du tableau de données $r$.



**Implémentation:**

- Dictionnaire de listes :

$$ I = \{c_1:(t_{11}, t_{12}, ...), c_2:(t_{21}, t_{22}, ...), ... \}$$

- à chaque classe est associée une *liste* de références

### Opérateurs d'agrégation

Une fois la partition effectuée, il est courant d'effectuer des **mesures** sur chaque partition obtenue

Les mesures sont réalisées à l'aide d'**opérateur d'agrégation** :

- comptage, somme, moyenne, ecart-type, max, min, ...
- (`count`, `sum`, `mean`, `avg`, `max`, `min`...)

**cas d'utilisation** :

- Quels sont les catégories de films/livres les plus fréquemment empruntés?
- A quelles heures de la journée la messagerie est-elle la plus sollicitée?
- Comment se répartissent géographiquement les utilisateurs de la messagerie?

### SQL

En SQL, l'opérateur d'agrégation est `GROUP BY`:

- partitionne les données à partir des valeurs de l'attribut mentionné
- il est possible de partitionner les données selon les valeurs de plusieurs attributs

Exemples de requêtes faisant appel aux fonctions d'agrégation :

*Nombre d'élèves par groupe de TD / par prépa d'origine etc..:*

```
SELECT groupe_TD , COUNT(num_eleve)
FROM Eleve
GROUP BY groupe_TD
```

*Donner les chiffres des ventes du magasin pour chaque mois de l'année*

```
SELECT mois, SUM(montant)
FROM Vente
GROUP BY mois
```

*Donner le nombre de ventes d'un montant > à 1000 euros pour les mois dont le chiffre d'affaires est supérieur à 10000 euros*

```
SELECT mois, COUNT(num_vente)
FROM Vente
GROUP BY mois
HAVING SUM(montant) >= 10000
```

*Tester les disparités salariales entre hommes et femmes*

```
SELECT sexe, avg( salaire )
FROM Employé
GROUP BY sexe
```

*Tester les disparités salariales selon le niveau d'éducation*

```
SELECT niveau_educatif, avg( salaire )
FROM Employé
GROUP BY niveau_éducatif
```

Pour aller plus loin :

- Une introduction très détaillée aux DataFrames (en Français)
- Introduction to Pandas (en anglais)

From:
https://wiki.centrale-med.fr/informatique/ - **WiKi informatique**

Permanent link:
**https://wiki.centrale-med.fr/informatique/tc_info:2026_cm_modeles?rev=1779998903**

Last update: **2026/05/28 22:08**