

# Exemple

Lorenzo Segoni

19 novembre 2025



# Table des matières

|       |  |   |
|-------|--|---|
| 0.1   | Bases de Données (BD) et SGBD . . . . .                | 1 |
| 0.1.1 | Introduction : La problématique des fichiers . . . . . | 1 |
| 0.1.2 | Définitions Fondamentales . . . . .                    | 1 |
| 0.1.3 | Objectifs d'une approche Base de Données . . . . .     | 2 |
| 0.1.4 | Les Fonctions du SGBD . . . . .                        | 2 |
| 0.2   | Le Modèle Conceptuel de Données (E/A) . . . . .        | 4 |
| 0.2.1 | Le Principe du Modèle E/A . . . . .                    | 4 |
| 0.2.2 | Les Composants Fondamentaux . . . . .                  | 4 |
| 0.2.3 | Les Associations (Les liens) . . . . .                 | 6 |
| 0.2.4 | Concepts Complémentaires . . . . .                     | 8 |
| 0.2.5 | Formalisation et Documentation . . . . .               | 8 |



## 0.1 Bases de Données (BD) et SGBD

### 0.1.1 Introduction : La problématique des fichiers

Avant l'invention des bases de données (années 1960), l'informatique reposait sur des systèmes de gestion de fichiers classiques. Cette méthode présentait des limites majeures qui ont conduit à la création des SGBD

#### Limite des systèmes de gestion de fichiers

Lorsqu'une application gère ses données via de simples fichiers, nous rencontrons trois problèmes fondamentaux :

- **La Redondance des données** : Les mêmes informations sont souvent répétées dans plusieurs fichiers pour différentes applications.  
*Conséquence* : Gaspillage d'espace et risque d'incohérence (si on modifie une info à un endroit mais pas à l'autre).
- **La Dépendance Programmes / Données** : La structure des données est "codée en dur" dans le programme.  
*Conséquence* : Si l'on change l'organisation physique d'un fichier, il faut réécrire tous les programmes qui l'utilisent. C'est une gestion complexe.
- **La Gestion des accès** : Il est difficile de permettre à plusieurs utilisateurs d'accéder et de modifier le même fichier en même temps sans créer de conflits.

### 0.1.2 Définitions Fondamentales

Il est crucial de ne pas confondre le contenu (la base) et le contenant/gestionnaire (le système).

#### Base de Données (BD)

**Définition 0.1.** Une Base de Données est une collection de données représentant des informations du monde réel.

Pour être qualifiée de BD, cette collection doit respecter quatre critères :

- **Cohérence et Structure** : Les données suivent un schéma logique défini.
- **Indépendance** : Les données existent indépendamment des applications qui les utilisent.
- **Non-redondance** : On évite de stocker deux fois la même information (redondance minimale).

- **Accessibilité** : Les données sont accessibles par plusieurs utilisateurs simultanément.

## Système de Gestion de Base de Données (SGBD)

**Définition 0.2.** Le SGBD est le logiciel qui sert d'interface entre les utilisateurs (ou applications) et la base de données.

Ses rôles principaux sont :

- La structuration des données.
- Le stockage physique.
- La mise à jour et la consultation.

**Exemples de domaines d'application** : Gestion d'entreprise, systèmes transactionnels (banques), e-commerce, bibliothèques numériques, etc.

### 0.1.3 Objectifs d'une approche Base de Données

L'utilisation d'un SGBD vise à résoudre les problèmes des systèmes de fichiers (vus en partie 1) en atteignant les objectifs suivants :

- **Indépendance Physique et Logique** : Le changement de la structure interne des données ou de leur stockage physique ne doit pas impacter les programmes. C'est l'objectif le plus important.
- **Manipulation aisée** : Permettre à des non-informaticiens d'interroger et de mettre à jour les données facilement.
- **Partage et Sécurité** : Plusieurs applications peuvent utiliser les mêmes données sans conflit.
- **Performance** : Garantir une efficacité d'accès (temps de réponse rapide) même avec de gros volumes de données.

### 0.1.4 Les Fonctions du SGBD

Pour atteindre ces objectifs, le SGBD offre quatre fonctions techniques majeures. Il est important de bien distinguer les deux langages (LDD et LMD).

#### Le Langage de Définition des Données (LDD)

Il permet de décrire la **structure** de la base (le squelette).

- **Rôle** : Définir les objets (tables), leurs attributs (colonnes), les liens entre eux et les contraintes.
- **Résultat** : On obtient le Schéma de la Base de Données.

### Le Langage de Manipulation des Données (LMD)

Il permet de gérer le **contenu** de la base (les données elles-mêmes).

- **Rôle** : Créer, modifier, supprimer ou consulter des données.
- **Outil** : C'est ici qu'intervient le langage SQL.

### Le Contrôle de l'intégrité

Le SGBD s'assure que les données respectent les règles définies (par le schéma ou le programme). Il empêche l'insertion de données aberrantes.

### La Sécurité de fonctionnement

Le SGBD gère les aspects critiques de l'exploitation :

- **Les Transactions et la journalisation** : Assurer que si une opération plante au milieu, on peut revenir en arrière (rollback) pour ne pas corrompre la base.
- **Les Accès concurrents** : Gérer plusieurs utilisateurs en même temps.
- **La Confidentialité** : Gérer les droits d'accès (qui a le droit de voir quoi).

## 0.2 Le Modèle Conceptuel de Données (E/A)

Le modèle Entité / Association

### 0.2.1 Le Principe du Modèle E/A

Avant de créer des tables dans l'ordinateur, il faut dessiner le schéma sur papier.

- **Origine** : Proposé par Peter Chen en 1976.
- **Objectif** : C'est une représentation graphique standardisée pour décrire les données d'un Système d'Information (SI).
- **Utilité** : Il sert de pont. Une fois le modèle E/A terminé, il est très facile de le traduire en tables SQL.

### 0.2.2 Les Composants Fondamentaux

Pour dessiner ce modèle, nous avons besoin de trois briques de base : l'Entité, l'Attribut et l'Identifiant.

#### L'Entité (L'objet)

**Définition 0.3.** Une entité est un objet (concret ou abstrait) à propos duquel on souhaite gérer des informations.

Il ne faut pas confondre le "moule" et l'objet créé :

1. **Type d'entité (Le moule)** : C'est la classe générale, le concept.  
*Exemple* : L'entité Étudiant, Client, Département.
2. **Occurrence d'entité (L'individu)** : C'est un élément précis, un individu spécifique qui appartient à ce type.  
*Exemple* : L'employé Alex Térieur ou Paul Auchon.

#### Les Attributs (Les détails)

**Définition 0.4.** Ce sont les propriétés qui décrivent une entité (ou une association).

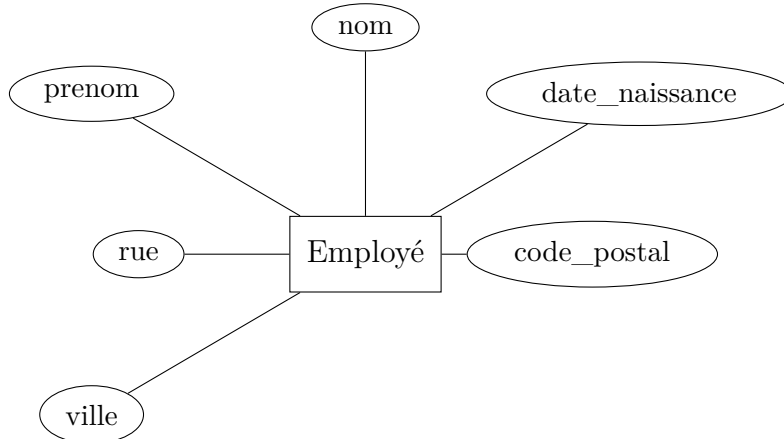
Chaque attribut possède :

1. **Un Nom** : (ex : Nom, Prix, Couleur).
2. **Un Domaine** : L'ensemble des valeurs possibles (ex : Entier, Réel positif, Chaîne de caractères, liste de choix Rouge, Vert, Bleu).



3. **Une Occurrence** : La valeur précise pour un individu (ex : "Rouge" est une occurrence de l'attribut **Couleur**).

**Représentation Graphique** : Dans le schéma, l'Entité est un Rectangle et les Attributs sont listés à l'intérieur (ou dans des bulles reliées au rectangle).



Que on peut représenter aussi :

| Entité : Employé  |
|-------------------|
| nom               |
| prénom            |
| date de naissance |
| rue               |
| code postal       |
| ville             |

TABLE 1 – Tableau Exemple

## L'Identifiant (La clé)

C'est le concept le plus important pour retrouver une info précise.

**Définition 0.5.** L'identifiant est l'ensemble minimal d'attributs qui permet de distinguer de façon unique chaque occurrence.

### Évolution de l'identifiant (Exemple du cours) :

1. **Mauvaise pratique** : Utiliser {Nom, Prénom, Date de naissance}. C'est lourd et il y a toujours un risque d'homonyme parfait.
2. **Bonne pratique (Identifiant artificiel)** : On ajoute un attribut dédié, souvent souligné dans le schéma.

*Exemple* : On ajoute Numero\_Employe.

Dans le schéma graphique, l'identifiant est toujours souligné.

| Entité : Employé         |
|--------------------------|
| <u>nom</u>               |
| <u>prénom</u>            |
| <u>date de naissance</u> |
| rue                      |
| code postal              |
| ville                    |

TABLE 2 – Tableau avec la mauvaise pratique

| Entité : Employé      |
|-----------------------|
| <u>Numero_Employé</u> |
| nom                   |
| prénom                |
| date de naissance     |
| rue                   |
| code postal           |
| ville                 |

TABLE 3 – Tableau avec la bonne pratique

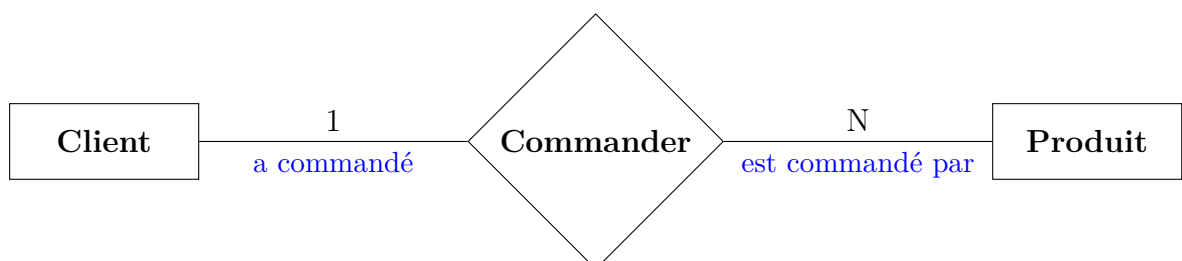
### 0.2.3 Les Associations (Les liens)

Les données ne vivent pas seules, elles sont reliées entre elles.

#### Définition

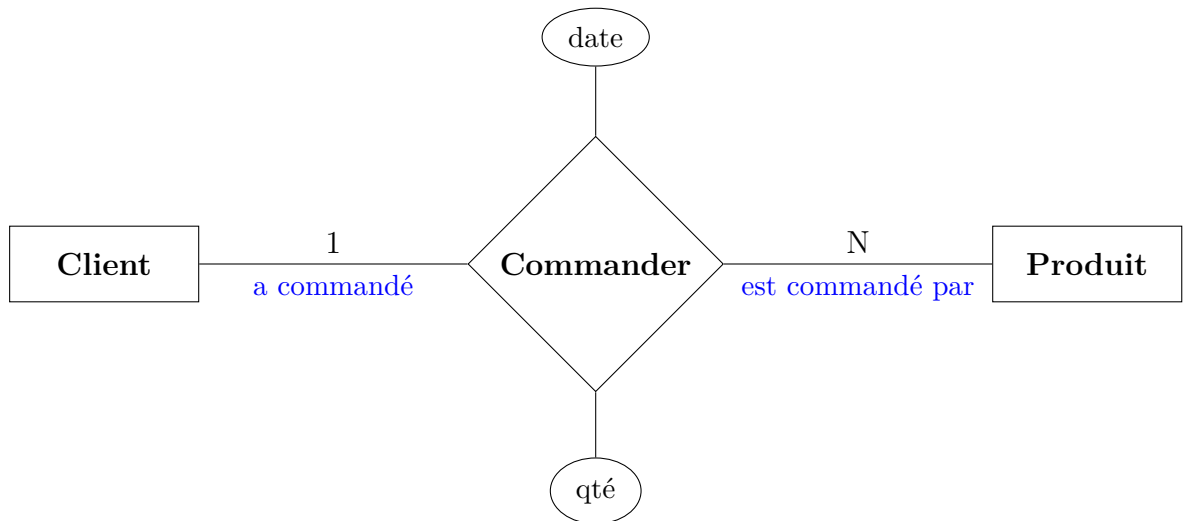
**Définition 0.6.** Une Association est un lien sémantique entre plusieurs entités. Elle est souvent représentée par un verbe.

— *Exemple :*



— **Attributs d'association :** Parfois, une donnée n'appartient ni à l'un, ni à l'autre, mais au lien lui-même.

*Exemple :* La **Quantité** (des produits commandé) et la **Date** (de la commande). Elles n'existent que parce qu'il y a une commande entre le client et le produit.

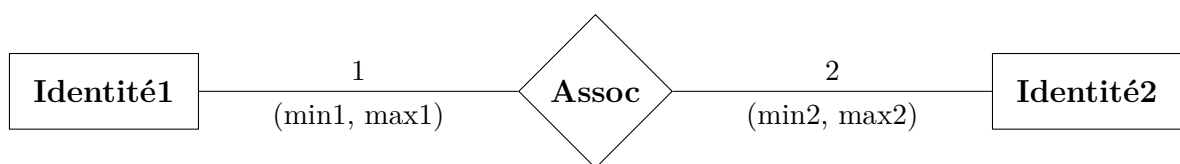


### Typologie des Associations

- **Binaire** : Relie 2 entités (le plus courant).
- **Ternaire** : Relie 3 entités.
- **Réflexive** : Une entité est reliée à elle-même (ex : Un employé est marié à un autre employé).

### Les Cardinalités (La règle du jeu)

**Définition 0.7.** Les cardinalités définissent les règles de quantité dans une association. Elles s'écrivent sous la forme (min, max) à côté de chaque entité.



### Comprendre le (min, max)

1. **Min (0 ou 1)** : Est-ce que l'entité est obligée de participer ?
  - 0 = Non (Optionnel).
  - 1 = Oui (Obligatoire).
2. **Max (1 ou N)** : Combien de fois maximum peut-elle participer ?
  - 1 = Une seule fois.
  - N (ou M) = Plusieurs fois (No limit).

## Les 3 grands types de relations

On classe les associations selon leur cardinalité **maximale** (le chiffre de droite) des deux côtés :

1. **Relation 1-1 (One-to-One) :**

Une occurrence de E est liée à **une seule** occurrence de F, et inversement.

2. **Relation 1-N (One-to-Many) :**

D'un côté, c'est unique (1), de l'autre c'est multiple (N).

*Exemple :* Un **Auteur** écrit plusieurs **Livres**, mais un **Livre** est écrit par un seul **Auteur** (dans ce modèle simplifié).

3. **Relation N-M (Many-to-Many) :**

Plusieurs des deux côtés.

*Exemple :* Un **Client** commande plusieurs **Produits**, et un **Produit** peut être commandé par plusieurs **Clients**.

## 0.2.4 Concepts Complémentaires

### Entités Faibles

**Définition 0.8.** Une entité qui ne possède pas d'identifiant propre.

- Elle ne peut exister que si elle est rattachée à une "Entité Forte".
- Son identifiant est composé de celui de l'entité forte + un identifiant partiel.

La cardinalité vers l'entité forte est toujours **(1,1)** (dépendance totale).

### Contraintes d'Intégrité (CI)

Ce sont des règles pour garantir que les données restent logiques.

- **CI Statiques :** Doivent être vraies tout le temps (ex : Le **Nom** est obligatoire, **Date naissance** < **Date mariage** , **Fax** attributs facultatifs, ... ).
- **CI Dynamiques :** Règles logiques sur des valeurs (ex : le **salaire** ne peut qu'augmenter).

## 0.2.5 Formalisation et Documentation

Pour qu'un projet de base de données soit valide, il ne suffit pas de faire un dessin (le schéma). Il faut respecter des **règles de complétude** : chaque objet dessiné doit être décrit textuellement de manière exhaustive.

L'ensemble de ces descriptions constitue l'**Univers du Discours** (le résumé synthétique de l'application).

### La Description des Objets (Entités et Associations)

Chaque élément graphique doit avoir sa fiche d'identité textuelle.

| Nom                   | Auteur  |
|-----------------------|---|
| Définition (Contexte) | Personne ayant écrit un livre référencé par l'éditeur |
| Liste d'attributs     | {nom, prénom, adresse}                                |
| Identifiant           | {nom, prénom}   |

TABLE 4 – Description d'une Entité

### Description d'une Association

Prenons l'association **Écriture** qui relie les auteurs aux livres.

| Nom                  | Ecriture  |
|----------------------|---|
| Définition           | L'écriture associe les livres à l'auteur qui les a écrits                 |
| Entités              | Auteur, Livre   |
| Rôles & Cardinalités | Un Auteur écrit 1 à $N$ Livres. Un Livre est écrit par 1 à 1 Auteur (1-1) |
| Attributs propres    | $\emptyset$ (Aucun attribut sur l'association elle-même)                  |

TABLE 5 – Description d'une Association

### Le Dictionnaire des Données

Le dictionnaire des données descend au niveau le plus fin : l'**attribut**. Il précise le format et les règles de chaque donnée.

| Nom                 | Ville auteur  |
|---------------------|---|
| Définition          | Nom de la ville où réside un auteur.                  |
| Structure           | Atomique (Mono-valué)                                 |
| Rôles / cardinalité | Chaîne de caractères alphabétiques.                   |
| Obligatoire ?       | Non. On peut créer un auteur sans connaître sa ville. |

TABLE 6 – Description d'un Attribut

### Les Contraintes d'Intégrité (CI)

Certaines règles logiques ne peuvent pas être dessinées sur le schéma. Il faut les écrire sous forme d'expressions logiques ou mathématiques.

|                                      |  |
|--------------------------------------|--|
| <b>Nom de la contrainte</b>          | Existence d'un mariage   |
| <b>Éléments concernés</b>            | Association : <u>Mariage</u> , Attribut : 'Âge de l'entité               |
| <b>Expression logique (La règle)</b> | Une occurrence de l'association <u>Mariage</u> n'est valide que si :Age( |

TABLE 7 – Contrainte sur une Association