# Formalisation des bases de données

Ludovic Deneuville

## Objectifs

Structurer une base de données pour qu'elle soit :

- efficace
- propre
- légère
- facile à maintenir

## Dépendances fonctionnelles

> **i** Définition
>
> Une dépendance fonctionnelle indique qu'une colonne (ou un ensemble de colonnes) détermine de manière unique une autre colonne :
>
> - $X \longrightarrow Y$
> - X détermine Y
> - Y dépend fonctionnellement de X

### Les axiomes d'Armstrong

| Nom | Description |
|---|---|
| Réflexivité | $(X, Y) \longrightarrow X$ |
| Augmentation | Si $X \longrightarrow Y$, alors $XZ \longrightarrow YZ$ |
| Transitivité | Si $X \longrightarrow Y$ et $Y \longrightarrow Z$ alors, $X \longrightarrow Z$ |

### Règles dérivées

| Nom | Description |
|---|---|
| Union | Si $X \longrightarrow Y$ et $X \longrightarrow Z$, alors $X \longrightarrow YZ$ |
| Décomposition | Si $X \longrightarrow YZ$, alors $X \longrightarrow Z$ et $X \longrightarrow Y$ |
| Pseudo-transitivité | Si $X \longrightarrow Y$ et $WY \longrightarrow Z$ alors $WX \longrightarrow Z$ |

### Deux Propriétés

> **i** Dépendance Fonctionnelle Élémentaire
>
> $X \longrightarrow Y$ est élémentaire s'il n'existe aucun sous-ensemble strict Z inclus dans X tel que $Z \longrightarrow Y$

**i** Dépendance Fonctionnelle Directe

X ⟶ Y est directe s'il n'existe pas de Z tel que X ⟶ Z et Z ⟶ Y

# Formes Normales

## Normalisation

**i** Définition

La normalisation vise à organiser les données :

- en éliminant les redondances et les incohérences
- en préservant les dépendances fonctionnelles
- sans perdre d'informations

## 1ère forme normale

**i** Définition

Une relation est 1NF si tous ses attributs sont atomiques.

- Une *date* est atomique
- Un *nom* est atomique
- Une *liste de clubs* n'est pas atomique
- Est-ce qu'un *adresse* est atomique ?

## 2e forme normale

**i** Définition

Une relation est 2NF si :

- elle est 1NF
- tout attribut non clé est en DF élémentaire avec la clé

Autrement dit, tous les attributs sont déterminés par la clé complète.

---

Exemple de relation non 2NF :

- participation(id_tournoi, id_joueuse, classement, nom_joueuse, nom_tournoi)
  - *nom_joueuse* dépend uniquement de *id_joueuse*
  - *nom_tournoi* dépend uniquement de *id_tournoi*

### 3e forme normale

> **Définition**
>
> Une relation est 3NF si :
>
> - elle est 2NF
> - tout attribut non clé est en DF directe avec la clé
>
> Autrement dit, un attribut ne peut pas dépendre fonctionnellement d'un autre attribut non clé.

---

Exemple de relation non 3NF :

- joueuse(id_joueuse, code_pays, nom_pays)
- id_joueuse ⟶ (code_pays, nom_pays)
- or code_pays ⟶ nom_pays donc DF non directe

> **Passer en 3NF**
>
> Créer une table *pays*
>
> - joueuse(id_joueuse, code_pays)
> - pays(code_pays, nom_pays)

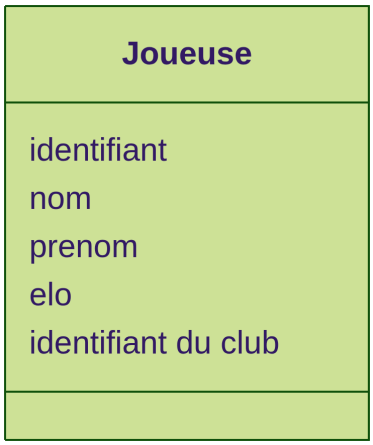
## Modélisation UML

### Unified Modeling Language

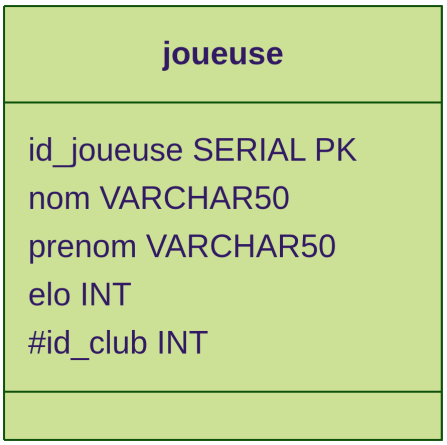
> **Définition**
>
> UML est un langage de modélisation graphique standardisé utilisé pour spécifier, visualiser, concevoir et documenter différents aspects des systèmes logiciels.
>
> Il offre une série de diagrammes pour représenter différents aspects d'un système.

### Modèles

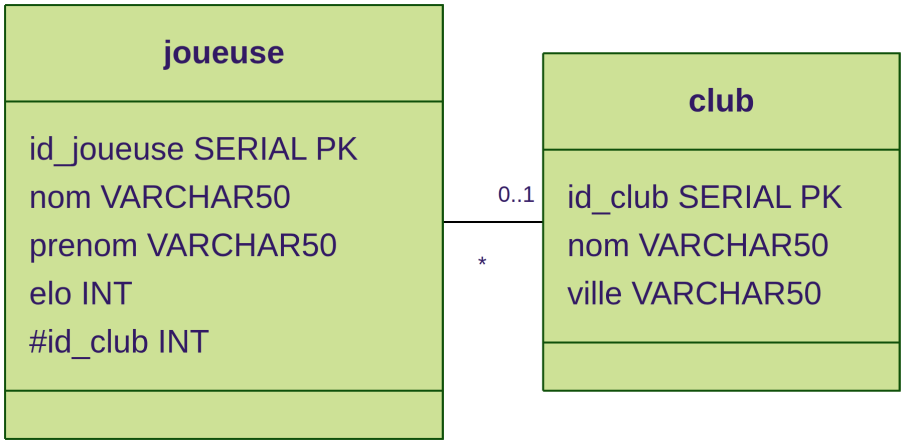
Modèle conceptuel



Modèle physique

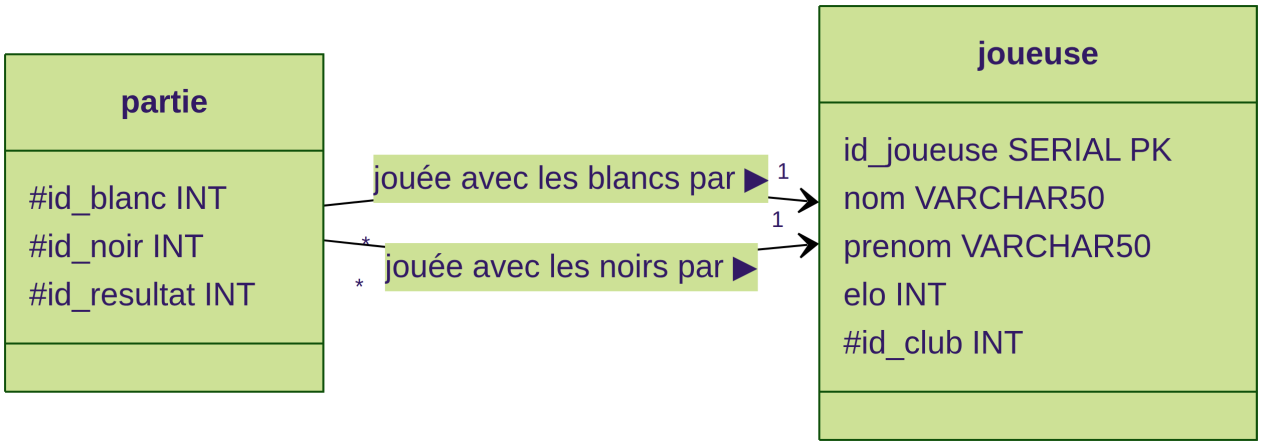



## Associations 1:n

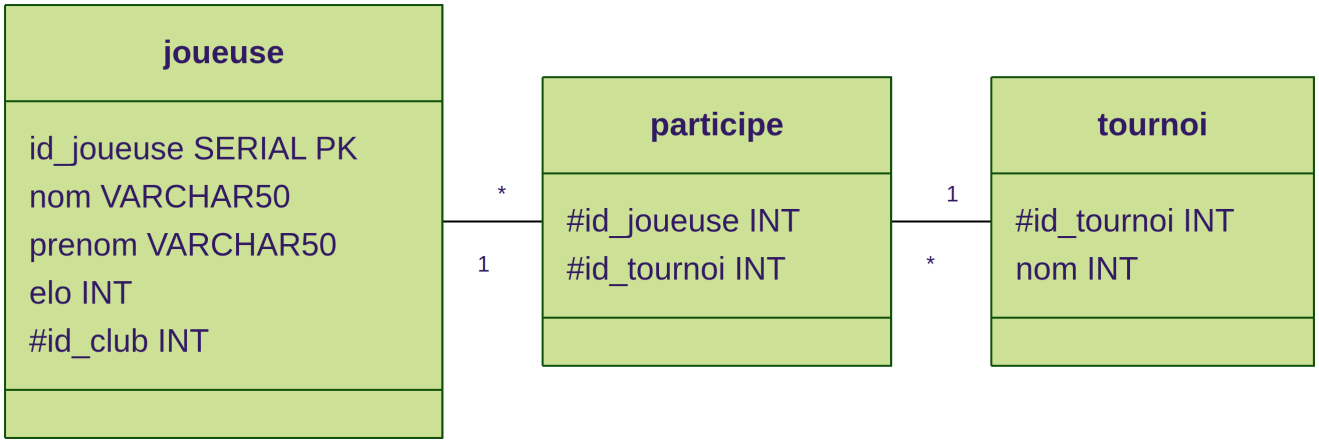



cf. Cours BDD, partie associations

## Associations multiples



## Associations n:n




## Outils

De nombreux outils existent pour faire des diagrammes UML, soit en mode graphique ou code, par exemple :

- https://app.diagrams.net/
- https://dbdiagram.io/d